



TrailsCommunity

Rapport de projet - Assistant Offroad

BELLANGER Stéphen MONNIER Ysée DESHORS Yann

Novembre 2016 - Décembre 2016

Table des matières

I Spécification des exigences logicielles	7
1 Introduction	8
1.1 Objectifs	8
1.2 Portée	8
1.3 Définitions	9
1.4 Références	9
1.5 Aperçu	9
1.6 Outils utilisés	9
2 Description générale	11
2.1 Perspective du produit	11
2.2 Fonctions du produit	11
2.3 Caractéristique de l'utilisateur	12
2.4 Contraintes	12
2.5 Hypothèses et dépendances	13
2.6 Répartition des besoins	13
3 Exigences particulières	14
3.1 Exigences externe de l'interface	14
3.2 Les interfaces utilisateurs	14
3.3 Interfaces matérielles	26
3.4 Interfaces logicielles	26
3.5 Interfaces de communication	26
4 Exigences fonctionnelles	27
4.1 Visiteur	27
4.1.1 Exigence fonctionnelle 1.1	27
4.1.2 Exigence fonctionnelle 1.2	27
4.1.3 Exigence fonctionnelle 1.3	27
4.1.4 Exigence fonctionnelle 1.4	28
4.2 Utilisateur	28
4.2.1 Exigence fonctionnelle 2.1	28
4.2.2 Exigence fonctionnelle 2.2	28
4.2.3 Exigence fonctionnelle 2.3	28
4.3 Participant	29
4.3.1 Exigence fonctionnelle 3.1	29

4.3.2	Exigence fonctionnelle 3.2	29
4.3.3	Exigence fonctionnelle 3.3	29
4.4	Organisateur	29
4.4.1	Exigence fonctionnelle 4.1	29
4.4.2	Exigence fonctionnelle 4.2	30
4.4.3	Exigence fonctionnelle 4.3	30
4.5	Application	30
4.5.1	Exigence fonctionnelle 5.1	30
4.5.2	Exigence fonctionnelle 5.2	31
4.6	IHM	31
4.6.1	Exigence fonctionnelle 6.1	31
4.6.2	Exigence fonctionnelle 6.2	31
4.6.3	Exigence fonctionnelle 6.3	31
4.6.4	Exigence fonctionnelle 6.4	32
4.6.5	Exigence fonctionnelle 6.5	32
4.6.6	Exigence fonctionnelle 6.6	32
4.7	Performances	33
4.7.1	Exigence fonctionnelle 7.1	33
4.7.2	Exigence fonctionnelle 7.2	33
4.7.3	Exigence fonctionnelle 7.3	33
4.8	Sécurité	33
4.8.1	Exigence fonctionnelle 7.1	33
4.8.2	Exigence fonctionnelle 7.2	34
4.8.3	Exigence fonctionnelle 7.3	34
4.8.4	Exigence fonctionnelle 7.4	34
5	Conception UML	35
5.1	Modélisation de l'axe statique	35
5.1.1	Diagramme de contexte statique	35
5.1.2	Diagramme des cas d'utilisation	37
5.1.3	Diagramme de classe	38
5.1.4	Diagramme entité association	42
5.2	Modélisation des axes fonctionnel et dynamique	42
5.2.1	Généralités	42
5.2.2	Traitement d'un cas d'utilisation - Créer une session	45
5.2.3	Traitement d'un cas d'utilisation - Rejoindre une session	46
5.3	Test	51
5.3.1	Test unitaire	51
5.3.2	Test UI	51
II	Serveur TrailsCommunity	54
1	Introduction	55
1.1	Conventions	55
1.2	Codes statut	55

2 Ressources	56
2.1 Authentification	56
2.1.1 Requête	56
2.1.2 Réponse	56
2.2 Inscription	57
2.2.1 Requête	57
2.2.2 Réponse	58
2.3 Ajout d'une session	58
2.3.1 Requête	58
2.3.2 Réponse	59
2.4 Liste des sessions	59
2.4.1 Requête	59
2.4.2 Réponse	60
2.5 Session	61
2.5.1 Requête	61
2.5.2 Réponse	62
2.6 Mis à jour d'une session	63
2.6.1 Requête	63
2.6.2 Réponse	63
2.7 Rejoindre une session	64
2.7.1 Requête	64
2.7.2 Réponse	64
2.8 Ajout d'un waypoint	64
2.8.1 Requête	64
2.8.2 Réponse	65
2.9 Ajout d'une coordonnée	65
2.9.1 Requête	65
2.9.2 Réponse	66
2.10 Envoyer un message	66
2.10.1 Requête	66
2.10.2 Réponse	66
2.11 Test	67
2.11.1 Test unitaire	67
III Manuel utilisateur	69
IV Déroulement du projet	79
0.1 Difficultés rencontrées	80
0.2 Perspectives d'évolution	80
0.3 Conclusion	80

V Annexe - Diagrammes UML	81
1 Diagrammes généraux	82
2 Modélisation serveur	84
3 Diagrammes de séquences	85
4 Diagrammes d'activités	91
5 Diagrammes d'états	96

Table des figures

3.1	Maquette de l'IHM pour la connexion	14
3.2	Maquette de l'IHM pour la création d'un compte utilisateur	15
3.3	Maquette de l'IHM pour l'affichage des différentes sessions	16
3.4	Maquette de l'IHM pour la modification des données utilisateurs	17
3.5	Maquette de l'IHM pour la création d'une session	18
3.6	Maquette de l'IHM pour l'affichage d'une session historique	19
3.7	Maquette de l'IHM pour l'affichage du Google map vide	20
3.8	Maquette de l'IHM pour l'affichage des différentes positions des participants et de réponse au partage d'un waypoint.	21
3.9	Maquette de l'IHM pour le chat d'une session courante	22
3.10	Maquette de l'IHM pour l'affichage des statistiques d'une session courante	23
3.11	Maquette de l'IHM pour la modification des données d'une session	24
3.12	Maquette de l'IHM pour l'affichage de l'événement d'accepter un nouveau waypoint	25
5.1	Diagramme de contexte statique	36
5.2	Diagramme des cas d'utilisation	37
5.3	Diagramme de classe du package root	38
5.4	Diagramme de classe du package session	39
5.5	Diagramme de classe du package sessions	40
5.6	Diagramme de classe du package models	41
5.7	Diagramme entité association	42
5.8	Diagramme de séquence authentification	43
5.9	Diagramme de séquence créer un compte utilisateur	44
5.10	Diagramme de séquence créer une session	45
5.11	Diagramme d'activité de la création d'une session	46
5.12	Diagramme de séquence rejoindre une session	47
5.13	Diagramme d'activité rejoindre une session	48
5.14	Diagramme de séquence de l'affichage d'une session	49
5.15	Diagramme d'activité de l'affichage d'une session	50
1	Capture d'écran de la page d'authentification	70
2	Capture d'écran pour la création d'un compte utilisateur	71
3	Diagramme de séquence de l'affichage de l'ensemble des sessions	72
4	Capture d'écran de l'affichage de la création d'une session	73
5	Capture d'écran de l'affichage de la déconnexion	74
6	Capture d'écran de l'affichage de la popup de l'entrée du mot de passe	75
7	Capture d'écran de l'affichage d'une session	76

8	Capture d'écran de l'affichage du chat	77
9	Capture d'écran de l'affichage des statistiques	78
1.1	Diagramme d'architecture de l'API	82
1.2	Diagramme de classe	83
2.1	Diagramme de séquence pour la vérification de la cohérence des données entre le client et le serveur	84
3.1	Diagramme de séquence de l'ajout d'un waypoint	86
3.2	Diagramme de séquence du chat	87
3.3	Diagramme de séquence du la modification du profile	88
3.4	Diagramme de séquence pour la sélection d'une session	89
3.5	Diagramme de séquence pour l'utilisation de la géolocalisation	89
3.6	Diagramme de séquence pour l'affichage de l'ensemble des sessions	90
4.1	Diagramme d'activité de l'ajout d'un waypoint	92
4.2	Diagramme d'activité du chat	93
4.3	Diagramme d'activité pour l'authentification	93
4.4	Diagramme d'activité pour la modification du profile	94
4.5	Diagramme d'activité pour la sélection d'une session	94
4.6	Diagramme d'activité pour l'utilisation de la géolocalisation	95
4.7	Diagramme d'activité pour l'affichage de l'ensemble des sessions	95
5.1	Diagramme d'état d'un acteur	96
5.2	Diagramme d'état d'une session	97

Première partie

Spécification des exigences logicielles

Chapitre 1

Introduction

Cette section fournit une vue d'ensemble de tout ce qui est inclus dans ce document SRS(Software Requirements Specification). De plus, l'objectif de ce document est détaillé et une liste de définitions est fournie.

1.1 Objectifs

L'objectif de ce document est de fournir une description détaillée des conditions requises pour l'application mobile TrailsCommunity. Il illustrera le but et la description complète du développement du système. Il expliquera également les contraintes du système, l'interface et les interactions avec les applications externes. Ce document est principalement destiné à être proposé à un client pour son approbation. C'est aussi une référence pour le développement de la première version du système pour l'équipe de développement.

1.2 Portée

TrailsCommunity est une application mobile GPS qui permet aux utilisateurs de connaître en temps réel le positionnement des autres participants d'une activité en pleine air. L'ensemble des activités sont regroupées en type qui seront limités à une liste pré-définie. L'application doit être gratuite, simple d'utilisation et ergonomique. L'organisateur de l'activité devra fournir deux adresses. Une pour le point de départ de l'application et la deuxième pour le point d'arrivée. La création des sessions se fera directement sur l'application mobile. En outre, le logiciel a besoin à la fois d'Internet et d'une connexion GPS pour récupérer et afficher les résultats. Tous les systèmes d'informations sont conservés dans une base de données, qui se trouve sur un serveur web. De plus, une base de données interne sera créée pour pouvoir stocker différentes données relatives aux sessions lorsque l'utilisateur utilise l'application en hors-ligne. Le logiciel interagit également avec l'API Google Maps qui permettra d'afficher et de tracer les différents parcours des utilisateurs. De plus, le téléphone doit aussi être composé d'un GPS pour pouvoir localiser l'utilisateur à tous moment. L'application a également la capacité de représenter à la fois des informations sommaires et détaillées des différentes statistiques de l'utilisateur courant, mais aussi des autres participants de la session en cours.

1.3 Définitions

session : une session est une activité disponible dans l'application.

visiteur : personne n'étant pas identifiée.

utilisateur : personne identifiée à l'application. Il possède les choix de créer ou de rejoindre une session.

organisateur : utilisateur ayant créé une session et pouvant la gérer.

participant : utilisateur qui a rejoint une session.

waypoint : désigne un point de la route à atteindre où doit avoir lieu un changement de cap.

1.4 Références

The Institute of Electrical and Electronic Engineer NY USA IEEE Recommended Practice for SRS,
<http://www.utdallas.edu/~chung/RE/IEEE830-1993.pdf>

Chalmers IEEE standards - SRS Example, 2010, http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf

Droid5 Informatics Pvt Ltd androidhive, 2016, <http://www.androidhive.info>

Oracle Java 8 Documentation, 2016, <https://docs.oracle.com/javase/8/docs/api/>

Google Android Documentation, 2016, <https://developer.android.com/guide>

James Britt Ruby Doc, 2016, <http://ruby-doc.org>

Rails Community Ruby On Rails Documentation, 2016, <http://rubyonrails.org>

1.5 Aperçu

Différentes techniques de spécification sont utilisées afin de préciser les exigences plus précisément pour différents publics. Cependant, le lecteur principal sera le correcteur du projet.

Le reste de ce document comprend cinq chapitres. Le second fournit une vue d'ensemble de la fonctionnalité du système et de l'interaction de l'application avec d'autres systèmes. En outre, le chapitre mentionne également les contraintes du système et les hypothèses sur le produit. Le troisième chapitre fournit la conception et l'ergonomie de l'application. Le quatrième chapitre concerne l'ensemble des exigences fonctionnelles de l'application. Elles sont triées en différentes parties. Le cinquième chapitre regroupe un ensemble de diagrammes UML qui sont primordiaux pour la compréhension, mais aussi pour le développement de l'application.

1.6 Outils utilisés

De nombreux outils seront utilisés tout au long du développement de l'application. En effet, l'équipe utilisera le logiciel TRELLO pour la gestion de projet et la répartition des tâches entre les différents membres. Dans un domaine plus précis, les développeurs ont mis en place un dépôt GIT qui permet de pouvoir centraliser l'ensemble du travail réalisé ainsi que de gérer les différentes

versions du produit. De plus, un outil d'intégration continue à été intégré pour suivre à chaque dépôt de code, la qualité de celui-ci.

Chapitre 2

Description générale

2.1 Perspective du produit

Ce système se compose de deux parties : une application mobile et un serveur web. L'application Android sera utilisée pour la gestion des sessions. Elle permettra d'afficher en fonction d'une session sélectionnée, sur une carte Google maps les différentes positions des participants.

Le serveur sera tant qu'à lui capable de gérer la récolte et la gestion des données envoyées par l'application. De plus, un mode hors-ligne sera disponible sur l'application. En effet, elle basculera sur une base de données interne qui stockera l'ensemble des données. Lorsque l'utilisateur se reconnectera à internet. L'ensemble des informations stockées en interne seront envoyées au serveur.

L'application mobile devra communiquer avec un GPS qui se situe au sein du mobile. Le GPS fournira les emplacements de l'utilisateur. Pour ce qui est des autres participants, leurs coordonnées seront stockées dans la base de données du serveur. De plus, des statistiques sur l'utilisateur courant ainsi qu'un chat seront disponibles pour permettre aux différents membres de pouvoir communiquer ensemble.

De manière transparente comme il s'agit d'un produit axé sur la récolte de données, il lui faudra une base de données. La communication avec la base de données se passera via internet. Cependant, comme expliqué précédemment, l'application possèdera une base de données interne pour gérer le mode hors-ligne.

2.2 Fonctions du produit

Avec l'application mobile, les utilisateurs pourront créer un compte utilisateur. Cette première étape va permettre de collecter différentes informations primordiales sur les utilisateurs.

Après la réalisation de cette première étape, et la connexion de ceux-ci, les utilisateurs pourront sélectionner des sessions. Le résultat sera basé sur les sessions que l'utilisateur sélectionne. Il existe 3 types de sessions. Les sessions actives, elles sont encore réalisable et ne sont pas encore clôturées.

Cela veut donc dire que leurs dates de fin ne sont pas dépassées. Dans ces sessions, un chat en temps réel est disponible permettant de communiquer avec tous les participants.

Les sessions clôturé dont l'utilisateur à participé. Elle se classe dans la section historique ce qui permet à l'utilisateur courant de pouvoir re visualiser ses anciennes statistiques.

Enfin les sessions dont l'utilisateur est le créateur. Il a donc les droits de modifications sur celle-ci.

L'application va permet après sélection d'une session, d'afficher une carte Google maps ainsi qu'un ensemble de points permettant de décrire le déplacement des différents utilisateurs actif de la session. De plus, les sessions pourront être directement créées depuis l'application.

2.3 Caractéristique de l'utilisateur

Il existe quartes types d'acteurs qui interagissent avec le système : le visiteur, l'utilisateur, le participant et l'organisateur. Chacun de ces quartes a une utilisation différente du système affin que chacun ait leurs propres exigences.

Les visiteurs de l'application mobile ne peuvent utiliser TrailsCommunity si ils ne sont pas connecté. Cela signifie que le visiteur doit être en mesure de pouvoir créer un compte utilisateur et ainsi de pouvoir se connecter.

Les visiteurs connectés peuvent alors visualiser l'ensemble des sessions actives mais aussi leur historique de session ainsi que les sessions dont ils sont organisateurs. De plus, ils peuvent modifier leurs coordonnées personnelles.

Les participants sont des acteurs qui ont rejoint une session active, ils peuvent alors ajouter un waypoint et le partager à l'ensemble des participants. Ils peuvent aussi communiquer via un chat en temps réel. De plus, ils peuvent consulter leurs statistiques en cours.

Enfin, les organisateurs sont des utilisateurs qui ont crée une nouvelle session. En effet, ce sont eux dont l'impulsion est venue pour organiser une activité. Ils vont pouvoir partager le mot de passe de la session pour pouvoir la rejoindre.

2.4 Contraintes

L'application mobile est limitée au système de navigation GPS du téléphone portable. Comme il existe plusieurs systèmes de fabricants de GPS, la précision n'est pas la même pour chacun d'entre eux. En outre, il peut y avoir une différence de navigation en fonction des téléphones.

La connexion internet est également une contrainte pour l'application. Puisqu'elle récupère les données de la base de données, il est crucial qu'il existe une connexion. De plus, en fonction du réseau, il est possible d'avoir une différence du temps de réception des données plus lente. Il existera des fonctionnalités hors-ligne qui permettront entre autres de pouvoir basculer sur un nouveau protocole pour que les utilisateurs puissent continuer à communiquer entre eux sur le chat : le protocole SMS. L'ensemble des données des sessions récoltées sont alors stockées dans une base de données interne du mobile. Lors de la récupération du réseau internet, l'application va alors recevoir l'ensemble des données collectées en dur et va reprendre son état d'envoi naturel.

2.5 Hypothèses et dépendances

Le produit sera toujours utilisé sur les téléphones mobiles Android qui ont assez de performances. Si le téléphone ne dispose pas de ressources matérielles suffisantes pour l'application. Elle peut ne pas fonctionner comme prévu ou même pas du tout. De plus, la version minimum d'Android doit être Jelly Belly (Version 16).

Une autre hypothèse est que les composants GPS de tous les téléphones fonctionnent de la même manière. Si les téléphones ont un système GPS est différent des normes, l'application doit être spécifiquement adaptée à chaque interface.

2.6 Répartition des besoins

Dans le cas où le projet est retardé, certaines exigences pourraient être transférées au prochaine version de l'application. Les exigences avec une priorité haute doivent être développées lors du rendu de la première version de l'application.

Chapitre 3

Exigences particulières

Cette section contient toutes les exigences fonctionnelles et de qualité du système. Il donne une description du système et de toutes ses caractéristiques.

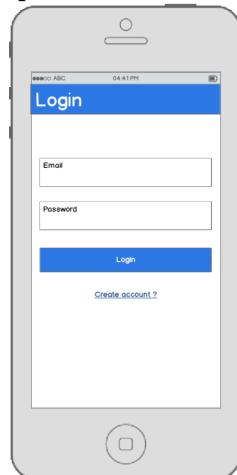
3.1 Exigences externe de l'interface

Cette section fournit une description détaillée de toutes les entrées et sorties du système. Il donne également une description des interfaces homme-machine et fournit des prototypes de base de l'interface pour l'utilisateur.

3.2 Les interfaces utilisateurs

Un visiteur de l'application mobile devrait voir la page d'ouverture de session quand il ouvre l'application. Pour la connexion au compte utilisateur, il lui est absolument nécessaire de posséder un compte utilisateur composé d'une adresse mail et du mot de passe associé.

FIGURE 3.1 – Maquette de l'IHM pour la connexion



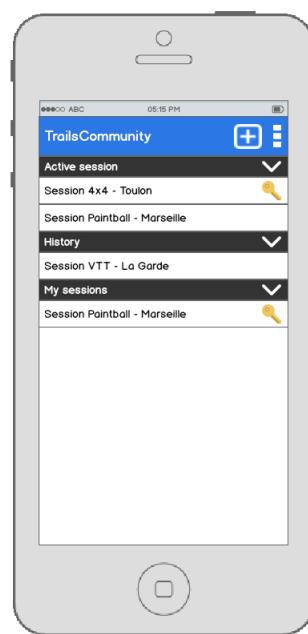
Pour la création d'un compte utilisateur, il est nécessaire de remplir l'ensemble des champs de l'interface. Toutefois, il sera possible de modifier son pseudonyme ainsi que son numéro de téléphone plus tard après sa connexion.

FIGURE 3.2 – Maquette de l'IHM pour la création d'un compte utilisateur



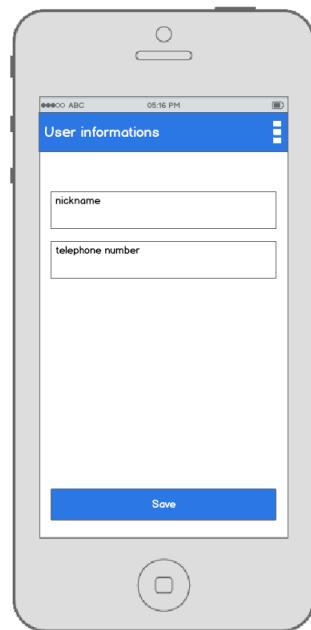
Après la connexion à un compte utilisateur, celui-ci peut alors choisir entre des sessions actives, les sessions dont il est l'organisateur ou des sessions clôturer qui sont dans la liste des historiques. L'icône plus permet de créer une nouvelle session. De plus, les boutons avec les 3 points permettent d'afficher une liste déroulante des différentes actions disponible sur cette interface. Les actions disponibles sont : la modification de ses données utilisatrice ainsi que la déconnexion.

FIGURE 3.3 – Maquette de l'IHM pour l'affichage des différentes sessions



Pour la modification des données utilisateurs, seulement le pseudonyme et le numéro de téléphone peuvent être modifié.

FIGURE 3.4 – Maquette de l'IHM pour la modification des données utilisateurs



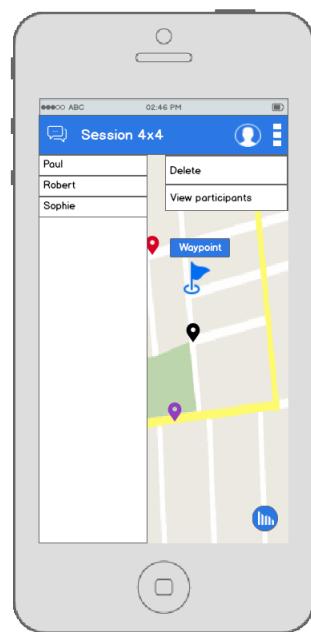
La création d'une session possède de nombreux champs texte obligatoire. Les adresses des lieux de départ et d'arrivée seront directement convertis en données GPS. De plus, lors du clique sur le champ texte de la date de départ, une boîte de dialogue apparaîtra avec un calendrier spécifique à Android. A noter que le champ du mot de passe ne sera connu que par l'organisateur de la session.

FIGURE 3.5 – Maquette de l'IHM pour la création d'une session



Lorsque l'utilisateur ouvre une session qui est dans sa liste de session historique. Cette interface apparaît avec des actions propres au session historique.

FIGURE 3.6 – Maquette de l'IHM pour l'affichage d'une session historique



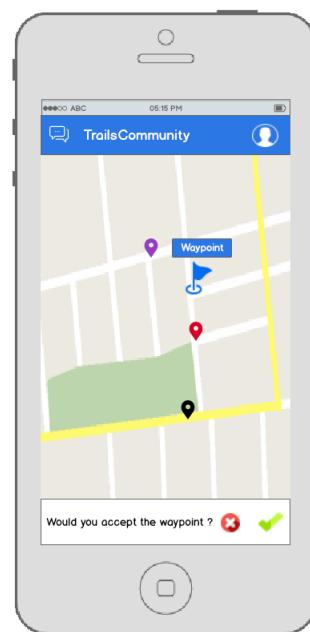
Cette maquette montre l'interface d'une session complètement vide. Seulement la Google maps y apparaît. Lorsque l'utilisateur a choisi une session dont il est l'organisateur, une action est disponible lorsque celui-ci clique sur les trois petits points : clôturer la session. Un pop-up de confirmation va alors apparaître pour qu'il puisse confirmer son action.

FIGURE 3.7 – Maquette de l'IHM pour l'affichage du Google map vide



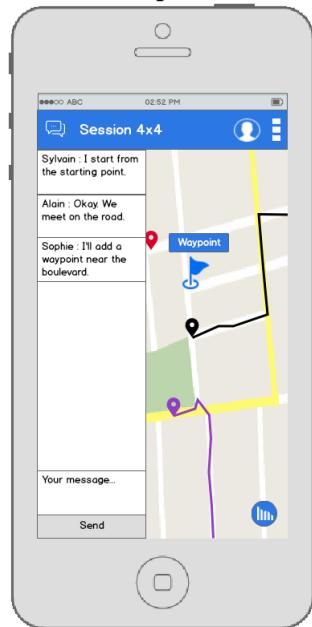
La maquette principale est l'affichage des différentes positions des participants de la session courante. Sur cette maquette, nous pouvons aussi voir comment l'utilisateur peut accepter ou non un waypoint partagé par un autre participant.

FIGURE 3.8 – Maquette de l'IHM pour l'affichage des différentes positions des participants et de réponse au partage d'un waypoint.



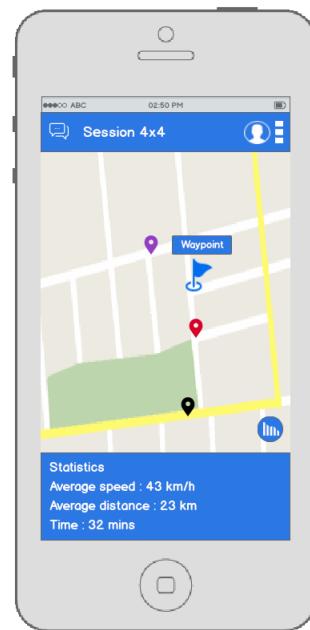
La maquette suivante présente le chat d'une session courante. La liste des messages se rafraîchi en temps réel.

FIGURE 3.9 – Maquette de l'IHM pour le chat d'une session courante



Cette maquette montre l'affichage des statistiques de la session courante. Les statistiques apparaissent lors du clique sur le bouton flottant en bas gauche.

FIGURE 3.10 – Maquette de l'IHM pour l'affichage des statistiques d'une session courante



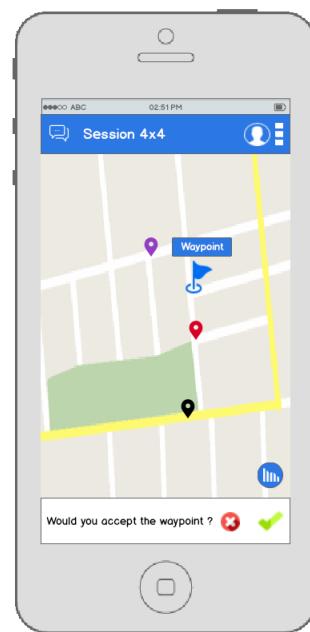
Cette maquette montre l'interface pour que l'organisateur puisse modifier l'ensemble des données de sa session.

FIGURE 3.11 – Maquette de l'IHM pour la modification des données d'une session



Cette maquette montre comment un waypoint est diffusé aux autres participants. L'utilisateur a alors le choix d'accepter ou de refuser que le waypoint diffusé apparaisse sur sa Google maps.

FIGURE 3.12 – Maquette de l'IHM pour l'affichage de l'événement d'accepter un nouveau waypoint



3.3 Interfaces matérielles

Puisque l'application mobile ne comporte pas de matériel désigné, il n'y a pas d'interfaces matérielles directes. Le GPS physique est géré directement par l'application et la connexion matérielle au serveur est géré par le système d'exploitation sous-jacent sur le téléphone mobile.

3.4 Interfaces logicielles

L'application mobile communique avec le GPS pour obtenir les informations géographiques. Pour la localisation de l'utilisateur et la représentation visuelle de celle-ci, l'utilisation du serveur est primordiale afin d'obtenir en temps réel les informations sur les participants. La communication entre la base de données et l'application se compose des opérations d'écriture et de lecture.

3.5 Interfaces de communication

La communication entre les différentes parties du système est importante car elles dépendent les unes des autres. Cependant, la manière dont la communication est réalisée n'est pas importante pour le système et est donc traitée par les systèmes d'exploitation pour l'application mobile.

Chapitre 4

Exigences fonctionnelles

Cette section comprend les exigences qui spécifient toutes les actions fondamentales du système logiciel.

4.1 Visiteur

4.1.1 Exigence fonctionnelle 1.1

ID : FR1

TITRE : Télécharger l'application mobile.

DESCRIPTION : Un utilisateur doit être en mesure de télécharger l'application mobile via le Google play store. L'application doit être libre de téléchargement.

BUT : Pour qu'un utilisateur puisse télécharger l'application mobile.

DÉPENDANCE : Aucune.

4.1.2 Exigence fonctionnelle 1.2

ID : FR2

TITRE : Télécharger et notifier les utilisateurs des nouvelles versions.

DESCRIPTION : Lorsqu'une nouvelle version/mise à jour est publiée, l'utilisateur doit les vérifier manuellement. Le téléchargement de la nouvelle version devrait se faire via le téléphone mobile de la même manière que le téléchargement de l'application mobile.

BUT : Pour que l'utilisateur puisse télécharger une nouvelle version.

DÉPENDANCE : FR1.

4.1.3 Exigence fonctionnelle 1.3

ID : FR3

TITRE : Enregistrement du visiteur sur l'application mobile.

DESCRIPTION : Étant donné qu'un utilisateur a téléchargé l'application mobile, l'utilisateur devrait pouvoir s'enregistrer via l'application mobile. L'utilisateur doit fournir en premier lieu son surnom puis vient en deuxième, adresse mail ainsi que son mot de passe, le code indicatif téléphonique et pour finir son numéro de téléphone.

BUT : Pour qu'un visiteur puisse s'inscrire sur l'application mobile.

DÉPENDANCE : FR1.

4.1.4 Exigence fonctionnelle 1.4

ID : FR4

TITRE : Connexion du visiteur sur l'application mobile.

DESCRIPTION : Étant donné qu'un visiteur s'est enregistré, il doit pouvoir se connecter à l'application mobile via son adresse mail et son mot de passe. Les informations d'ouverture de session seront stockées dans la base de données. Le mot de passe sera bien sûr crypté côté serveur.

BUT : Pour qu'un visiteur puisse se connecter sur l'application mobile.

DÉPENDANCE : FR1, FR3.

4.2 Utilisateur

4.2.1 Exigence fonctionnelle 2.1

ID : FR5

TITRE : Un utilisateur peut rejoindre une session.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il a le choix entre 3 types de sessions. Le premier type est les sessions actives, elles sont soit privée ou publique. Si elles sont privées, l'utilisateur doit connaître le mot de passe pour les rejoindre. Ensuite, le deuxième type est les sessions dont il a déjà participé, cela fonctionne comme un historique. La dernière est les sessions que l'utilisateur a créées et dont il est, bien sûr, l'organisateur.

BUT : Pour qu'un organisateur puisse rejoindre une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4.

4.2.2 Exigence fonctionnelle 2.2

ID : FR6

TITRE : Un utilisateur peut modifier ses données utilisateurs.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il peut modifier ses données utilisateurs. Les champs modifiables sont son pseudonyme, son numéro de téléphone. Les informations des données utilisatrices seront stockées dans la base de données. Dans le cas où celui-ci les modifie alors qu'il participe à une session. Les autres utilisateurs sont alors notifiés si l'utilisateur modifie son pseudo. La notification apparaîtra alors dans le chat.

BUT : Pour qu'un utilisateur puisse modifier ses données utilisateur sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4.

4.2.3 Exigence fonctionnelle 2.3

ID : FR7

TITRE : Un utilisateur peut se déconnecter de l'application.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, il doit avoir le choix, à tous moment de pouvoir se déconnecter de l'application. Il sera alors redirigé vers la page de connexion.

BUT : Pour qu'un organisateur puisse se déconnecter de l'application mobile.

DÉPENDANCE : FR1, FR3.

4.3 Participant

4.3.1 Exigence fonctionnelle 3.1

ID : FR8

TITRE : Un participant peut ajouter un waypoint dans une session active.

DESCRIPTION : Étant donné qu'un participant s'est connecté et à rejoint une session active, il peut ajouter un waypoint. Lorsque l'utilisateur maintient longuement son doigt sur l'écran de son téléphone, un marker est alors ajouté. Le waypoint est alors diffusé à tous les autres participants. Les autres utilisateurs ont alors le choix d'accepter ou de décliner le partage.

BUT : Pour qu'un participant puisse ajouter un waypoint dans une session active.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.3.2 Exigence fonctionnelle 3.2

ID : FR9

TITRE : Un participant peut communiquer via un système de chat en temps réel à la session courante.

DESCRIPTION : Étant donné qu'un participant s'est connecté et à rejoint une session active, il peut alors communiquer avec l'ensemble des utilisateurs participant à la session. Il lui suffit d'entrer son message et de l'envoyer. Il sera alors diffusé aux autres participants. De plus, si l'application est hors-ligne. Les messages seront alors envoyés en protocole SMS et stockés dans la base interne pour qu'ils puissent être envoyés au serveur ultérieurement.

BUT : Pour qu'un participant puisse communiquer via un système de chat en temps réel à la session courante.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.3.3 Exigence fonctionnelle 3.3

ID : FR10

TITRE : Un participant peut visualiser ses statistiques de la session active.

DESCRIPTION : Étant donné qu'un participant s'est connecté et à rejoint une session active, il peut alors visualiser l'ensemble de ses statistiques. Celle-ci décrit différentes données comme la distance totale parcourue, la vitesse moyenne du participant ou alors le nombre de waypoint ajoutés.

BUT : Pour qu'un participant puisse visualiser ses statistiques de la session active

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.4 Organisateur

4.4.1 Exigence fonctionnelle 4.1

ID : FR11

TITRE : Un organisateur peut créer une session.

DESCRIPTION : Étant donné qu'un organisateur s'est connecté, il doit pouvoir créer une nouvelle session dans l'application. Les informations de la nouvelle session créée seront stockées dans la base de données du serveur distant. L'organisateur doit fournir le lieu de départ ainsi que le lieu d'arrivée. Le format d'une adresse lui sera demandé. L'application convertira cette adresse en coordonnées GPS. De plus, il devra indiquer le type d'activité ainsi que la date de départ. Pour finir, l'organisateur aura la possibilité de verrouiller la session par un mot de passe qu'il divulguera à l'ensemble des participants.

BUT : Pour qu'un organisateur puisse créer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4.

4.4.2 Exigence fonctionnelle 4.2

ID : FR12

TITRE : Un organisateur peut gérer ses sessions.

DESCRIPTION : Étant donné qu'un organisateur s'est connecté, il doit pouvoir gérer ses sessions. Tous les champs remplis précédemment lors de la création de la session sont modifiables. Les informations de la session seront stockées dans la base de données.

BUT : Pour qu'un organisateur puisse gérer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.4.3 Exigence fonctionnelle 4.3

ID : FR13

TITRE : Un organisateur peut clôturer ses sessions.

DESCRIPTION : Étant donné qu'un organisateur s'est connecté et à sélectionné une de ses sessions créées précédemment. Il doit pouvoir clôturer la session sélectionnée. Celle-ci n'est alors plus présente dans les sessions actives de l'application sauf pour les participants où la session est alors disponible dans leurs historiques. Cette action est disponible dans le menu déroulant des trois petits points (voir maquette 3.7). Une boîte de dialogue de confirmation va alors apparaître pour qu'il puisse confirmer ou annuler son action.

BUT : Pour qu'un organisateur puisse clôturer une session sur l'application mobile.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.5 Application

4.5.1 Exigence fonctionnelle 5.1

ID : FR14

TITRE : L'application doit pouvoir enregistrer un parcours effectué sans connexion internet.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, a rejoint une session active et qu'il est déconnecté du réseau internet, l'application doit pouvoir gérer le mode hors ligne. Une base de données interne à l'application va alors stocker l'ensemble des données. Lorsque l'utilisateur sera de nouveau connecté, les informations seront alors synchronisées avec le serveur.

BUT : Pour que l'application puisse pouvoir enregistrer un parcours effectué sans connexion internet.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.5.2 Exigence fonctionnelle 5.2

ID : FR15

TITRE : Le chat de l'application doit pouvoir fonctionner en mode hors ligne.

DESCRIPTION : Étant donné qu'un utilisateur s'est connecté, à rejoint une session active et qu'il est déconnecté du réseau internet, l'application doit pouvoir gérer le mode hors ligne du chat. L'application va alors basculer sur le protocole SMS pour que les participants puissent continuer de communiquer ensemble. De plus, l'ensemble des messages seront stockés en internet et synchronisés avec le serveur lorsque l'utilisateur sera de nouveau en ligne.

BUT : Pour que le chat de l'application puisse fonctionner en mode hors ligne.

DÉPENDANCE : FR1, FR3, FR4, FR5.

4.6 IHM

4.6.1 Exigence fonctionnelle 6.1

ID : FR16

TITRE : Le système doit pouvoir afficher une carte Google maps interactive.

DESCRIPTION : Le système doit être capable d'afficher une carte Google maps interactive. En effet, cette carte doit pouvoir posséder les fonctionnalités de zoom ainsi qu'une navigation tactile et ergonomique.

BUT : Pour que le système puisse afficher une carte Google maps interactive.

DÉPENDANCE : Aucune.

4.6.2 Exigence fonctionnelle 6.2

ID : FR17

TITRE : Le système doit pouvoir ajouter un ou plusieurs markers sur une carte Google maps.

DESCRIPTION : Étant donné que le système a affiché une carte Google maps interactive, il doit permettre de pouvoir ajouter un ou plusieurs markers sur celle-ci. En effet ces markers peuvent posséder un titre interactif qui, lors d'une appui, peuvent afficher une multitude d'informations supplémentaires sur le marker associé.

BUT : Pour que le système puisse ajouter un ou plusieurs markers sur une carte Google maps.

DÉPENDANCE : F16.

4.6.3 Exigence fonctionnelle 6.3

ID : FR18

TITRE : Le système doit pouvoir afficher un ou plusieurs parcours sur une carte Google maps.

DESCRIPTION : Étant donné que le système a affiché une carte Google maps interactive, il doit permettre de pouvoir afficher un ou plusieurs parcours sur celle-ci. En effet, ces chemins

possèdent un marker de début et un marker de fin pour signaler aux autres participants où se situe l'utilisateur. Ce parcours possèdera une couleur unique dans la session active.

BUT : Pour que le système puisse afficher un ou plusieurs parcours sur une carte Google map.

DÉPENDANCE : F16, F17.

4.6.4 Exigence fonctionnelle 6.4

ID : FR19

TITRE : Le système doit pouvoir afficher la géolocalisation des participants d'une session active sur une carte Google maps.

DESCRIPTION : Étant donné que le système a affiché une carte Google maps interactive, il doit permettre de pouvoir afficher la géolocalisation des participants d'une session active sur celle-ci. En effet, cette géolocalisation est signalée par un marker au bout du chemin associé aux participants. Cette géolocalisation est réalisée par le GPS interne du téléphone portable. De plus, la précision de celle-ci dépendra totalement du système d'exploitation et du matériel du téléphone portable de l'utilisateur.

BUT : Pour que le système puisse afficher la géolocalisation des participants d'une session active sur une carte Google maps.

DÉPENDANCE : F16.

4.6.5 Exigence fonctionnelle 6.5

ID : FR20

TITRE : Le système doit pouvoir afficher différentes unités de mesure des statistiques en fonction des activités.

DESCRIPTION : Le système doit pouvoir afficher différentes unités de mesures en fonction du type d'activité. En effet, les mesures sont différentes pour une activité se réalisant avec une voiture et une activité se réalisant en bateau ou à pied. Le système va alors déterminer en fonction du type d'activité choisi l'unité de mesure le plus adéquat.

BUT : Pour que le système puisse afficher différentes unités de mesure des statistiques en fonction des activités.

DÉPENDANCE : Aucune.

4.6.6 Exigence fonctionnelle 6.6

ID : FR21

TITRE : Le système doit pouvoir gérer la gestion un mode multi-session.

DESCRIPTION : Le système doit permettre un système de multi-session. Il permettra à l'utilisateur de pouvoir utiliser son compte sur différent mobile.

BUT : Pour que le système puisse gérer un mode multi-session.

DÉPENDANCE : Aucune.

4.7 Performances

4.7.1 Exigence fonctionnelle 7.1

ID : FR22

TITRE : Le système doit afficher rapidement l'ensemble des sessions dans les listes déroulantes.

DESCRIPTION : Le système doit posséder un niveau de performance élevé pour l'affichage des différentes sessions envoyé par le serveur. En effet, la rapidité d'affichage doit être rapide et aucune latence ne doit être observée par l'utilisateur. De plus, un maximum de 50 sessions actives sera affiché.

BUT : Pour que le système puisse afficher rapidement l'ensemble des sessions dans les listes déroulantes.

DÉPENDANCE : Aucune.

4.7.2 Exigence fonctionnelle 7.2

ID : FR23

TITRE : Le système doit afficher rapidement l'ensemble des différents éléments affichés sur la carte Google maps interactive.

DESCRIPTION : Le système doit posséder un niveau de performance élevé pour l'affichage des différents éléments affichés sur la carte Google maps. En effet, aucune latence ne doit être observée par l'utilisateur lors de la réception de coordonnées GPS ou l'affichage d'un nouveau waypoint. De plus, l'affichage des parcours de l'ensemble des participants d'une session active ne doit pas entraîner une surcharge du serveur.

BUT : Pour que le système puisse afficher rapidement l'ensemble des différents éléments affichés sur la carte Google maps interactive.

DÉPENDANCE : Aucune.

4.7.3 Exigence fonctionnelle 7.3

ID : FR24

TITRE : Le serveur doit posséder un temps de réponse inférieur à une seconde.

DESCRIPTION : Le serveur doit posséder un temps de réponse inférieur à une seconde. Ce temps est un temps de performance idéal cependant, le temps de réponse doit être au minimum inférieur à deux secondes.

BUT : Pour que le serveur doive posséder un temps de réponse inférieur à une seconde.

DÉPENDANCE : Aucune.

4.8 Sécurité

4.8.1 Exigence fonctionnelle 7.1

ID : FR25

ESSENTIEL : La sécurité de création de compte pour les utilisateurs du système.

DESCRIPTION : Si un utilisateur veut créer un compte et que le nom de l'utilisateur souhaité est déjà utilisé, l'utilisateur va être invité à choisir un nom d'utilisateur différent.

NIVEAU : Haute.

4.8.2 Exigence fonctionnelle 7.2

ID : FR26

ESSENTIEL : La sécurité de la communication entre le système et le serveur.

DESCRIPTION : Les messages doivent être chiffrés pour les communications de connexion, de sorte que d'autres utilisateurs ne peuvent pas obtenir le nom d'utilisateur et le mot de passe à partir de ces messages.

NIVEAU : Haute.

4.8.3 Exigence fonctionnelle 7.3

ID : FR27

ESSENTIEL : Le nombre de requêtes par seconde côté serveur.

DESCRIPTION : Le serveur peut ajouter une limitation de requête par seconde pour un utilisateur. Cela permet d'éviter les attaques de brute forces et autres attaques d'injection de code.

NIVEAU : Haute.

4.8.4 Exigence fonctionnelle 7.4

ID : FR28

ESSENTIEL : Utilisation d'un token pour la communication client-serveur.

DESCRIPTION : Les échanges entre client-serveur se font grâce à un token généré par le serveur. Ce token est unique et propre à chaque utilisateur. L'utilisation de ce token permet de sécuriser les ressources du serveur. À chaque requête, le token doit être présent dans l'en-tête. S'il n'y a pas de token, le serveur renvoie une réponse comme quoi l'utilisateur n'est pas autorisé à accéder à cette ressource.

NIVEAU : Haute.

Chapitre 5

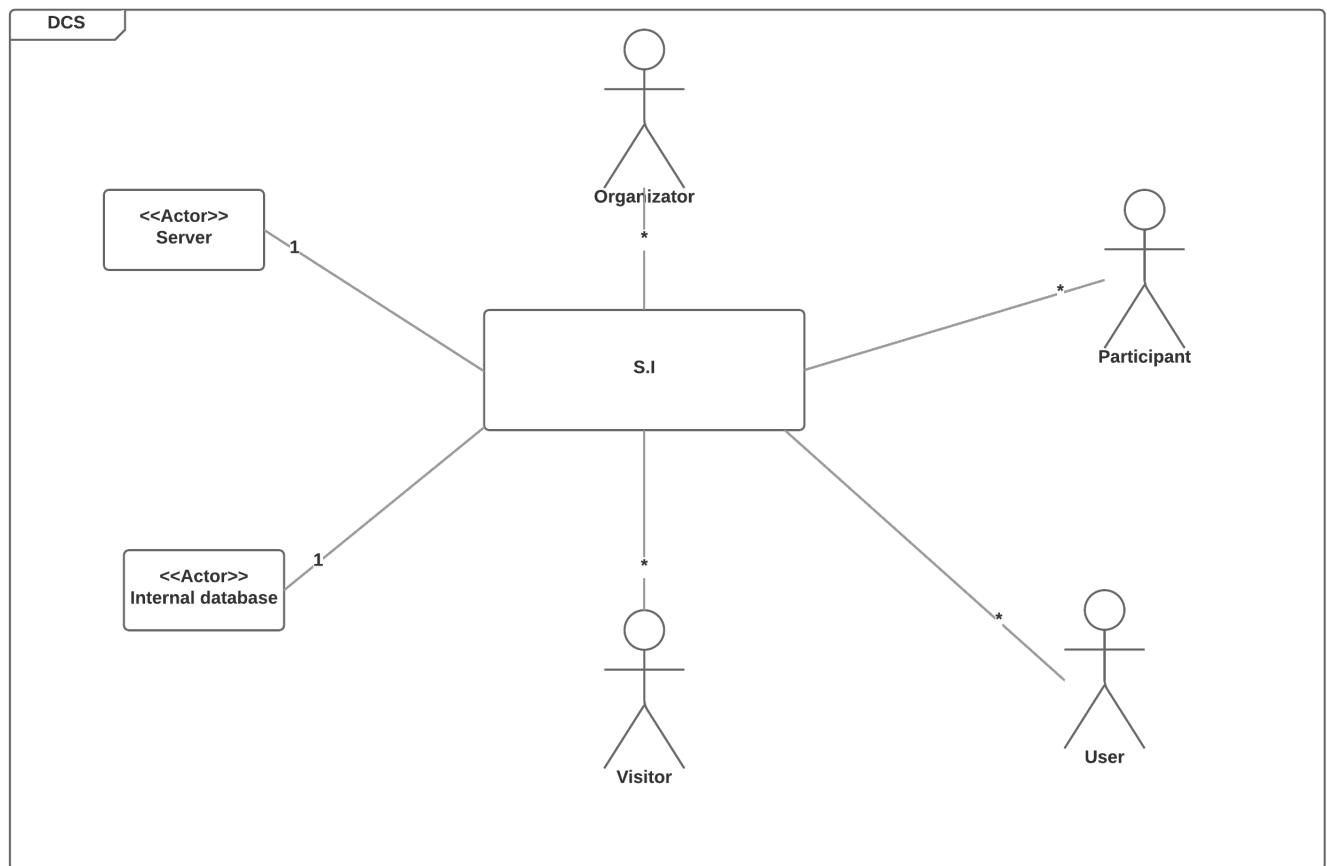
Conception UML

5.1 Modélisation de l'axe statique

5.1.1 Diagramme de contexte statique

Nous distinguons quatre types d'acteurs : un visiteur, un utilisateur, un participant et un organisateur. Chaque acteur aura des utilisations de l'application mobile complètement distincts.

FIGURE 5.1 – Diagramme de contexte statique

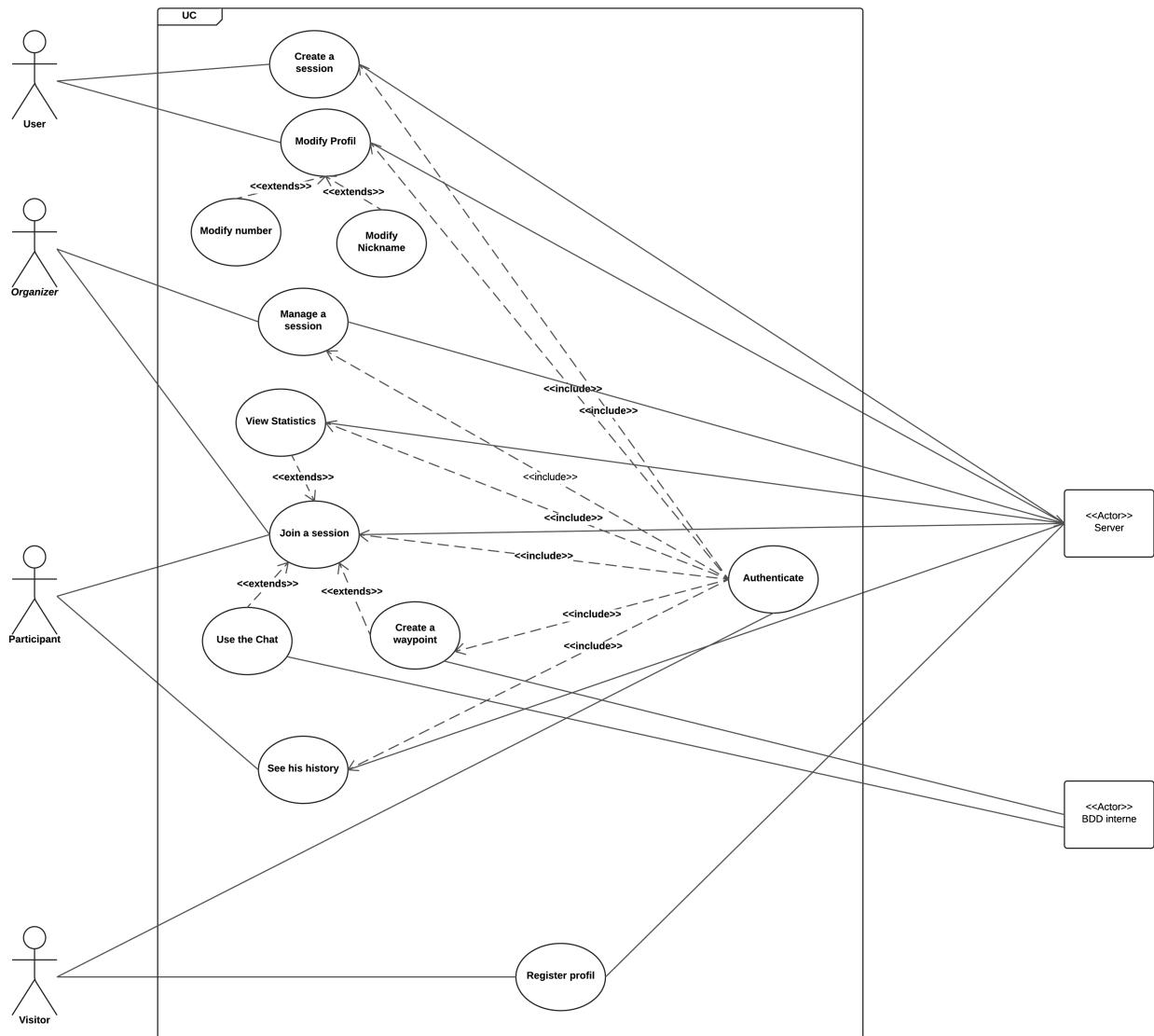


L'accès aux données nécessaires au fonctionnement de l'application se fera par le biais d'un serveur codé en Ruby dont le fonctionnement est détaillé en annexe. C'est ce serveur qui communiquera avec la base de données qui stocke l'ensemble des informations des sessions.

5.1.2 Diagramme des cas d'utilisation

L'ensemble des cas d'usages excepté la création d'un compte utilisateur requière une authentification. Le statut de l'acteur diffère en fonction des actions qu'il réalise sur l'application.

FIGURE 5.2 – Diagramme des cas d'utilisation



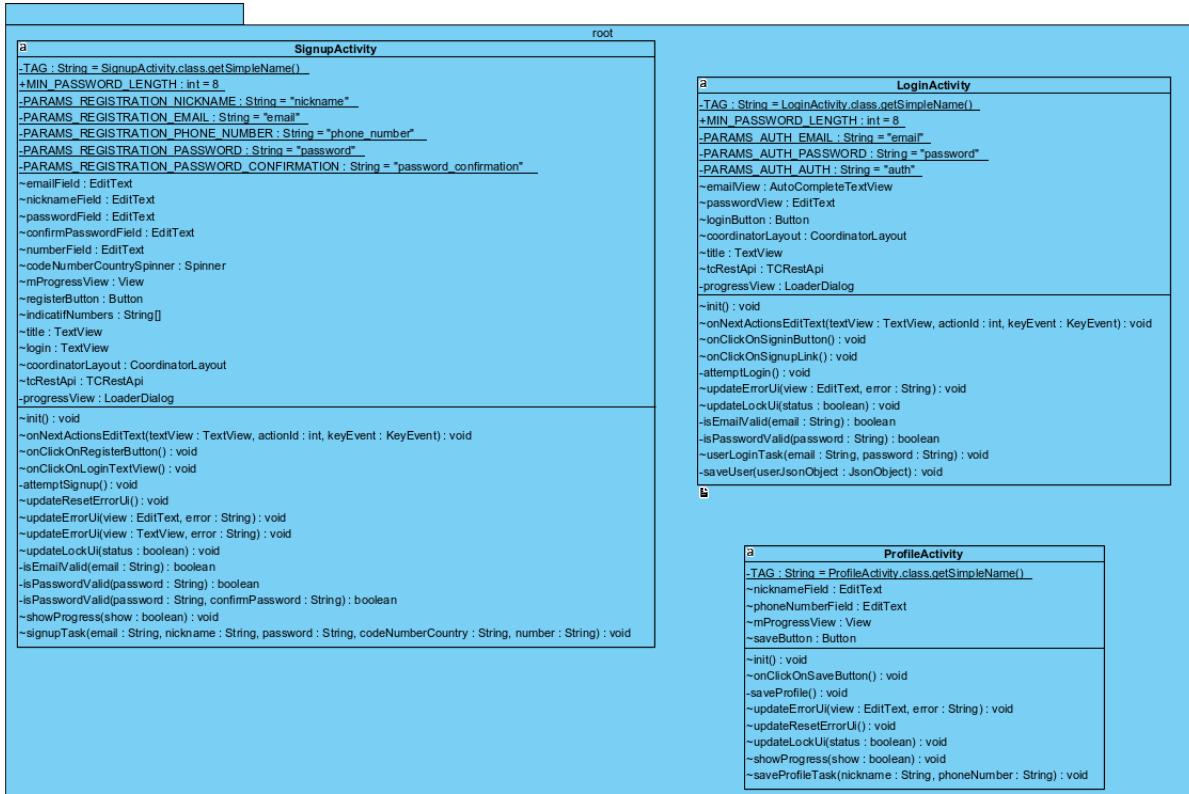
La priorité des cas d'utilisation est détaillée dans un tableau des priorités plus bas dans le rapport. La principale fonctionnalité de l'application est d'assurer la possibilité de créer une session d'une activité quelconque et d'y participer.

Use case	Priority	Risk
Create a session	High	High
Manage a session	High	High
Join a session	High	Mean
Create a waypoint	Mean	Mean
Authentification	Mean	Mean
Use the chat	Low	Low
See statistics	Low	Low
See his history	Low	Low
Modify profil	Low	Low

5.1.3 Diagramme de classe

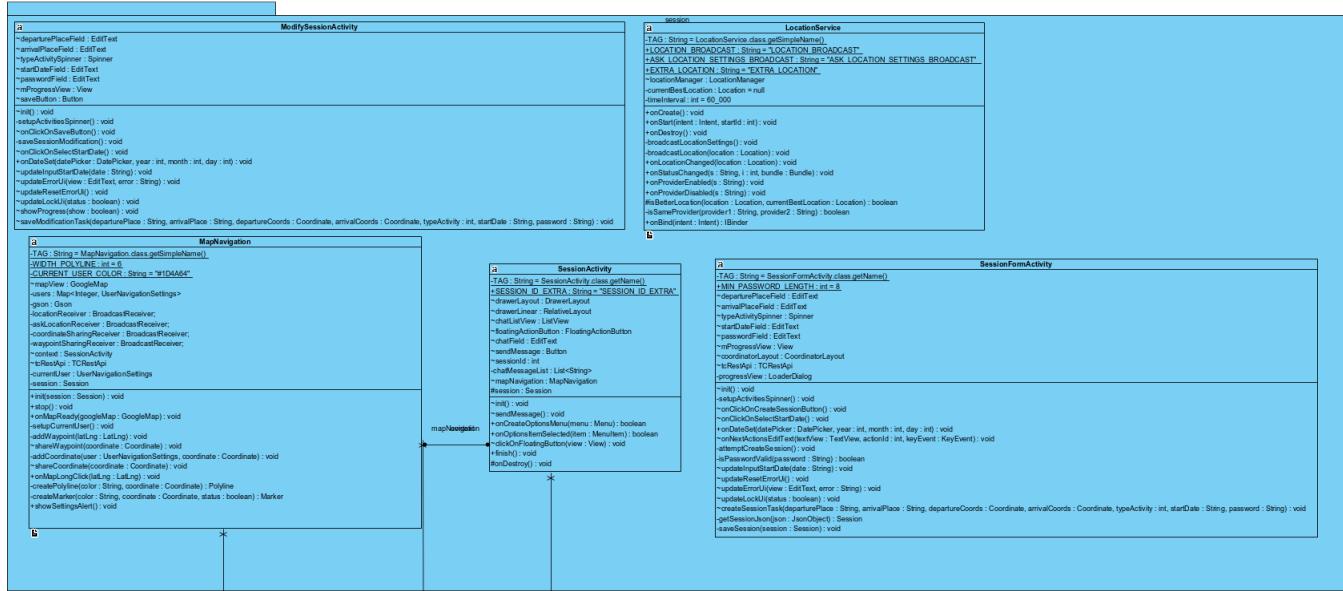
5.1.3.1 Package root

FIGURE 5.3 – Diagramme de classe du package root



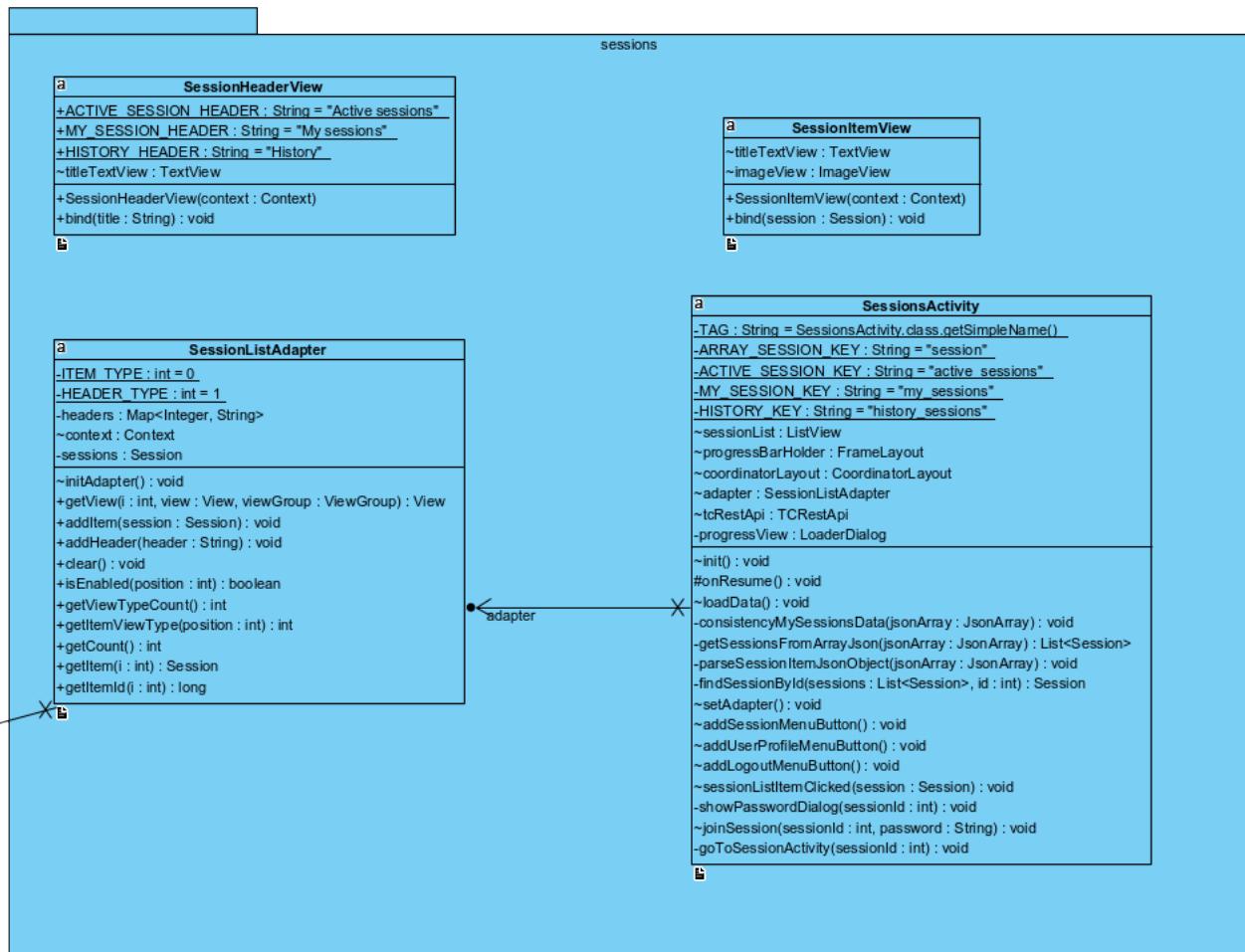
5.1.3.2 Package session

FIGURE 5.4 – Diagramme de classe du package session



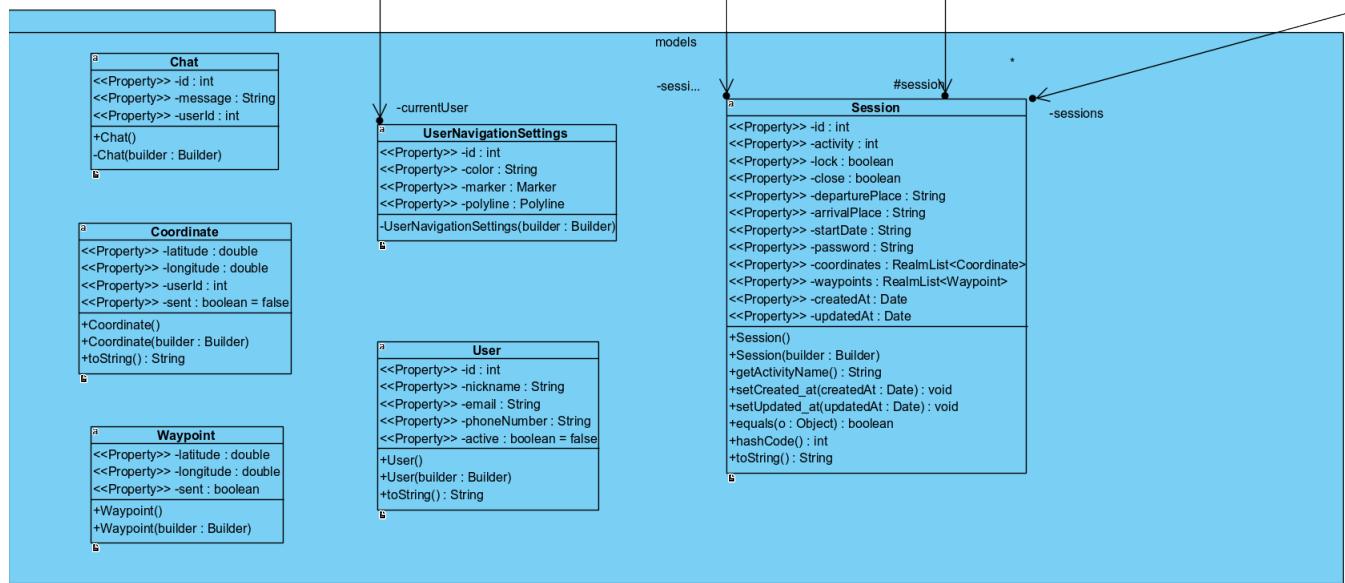
5.1.3.3 Package sessions

FIGURE 5.5 – Diagramme de classe du package sessions



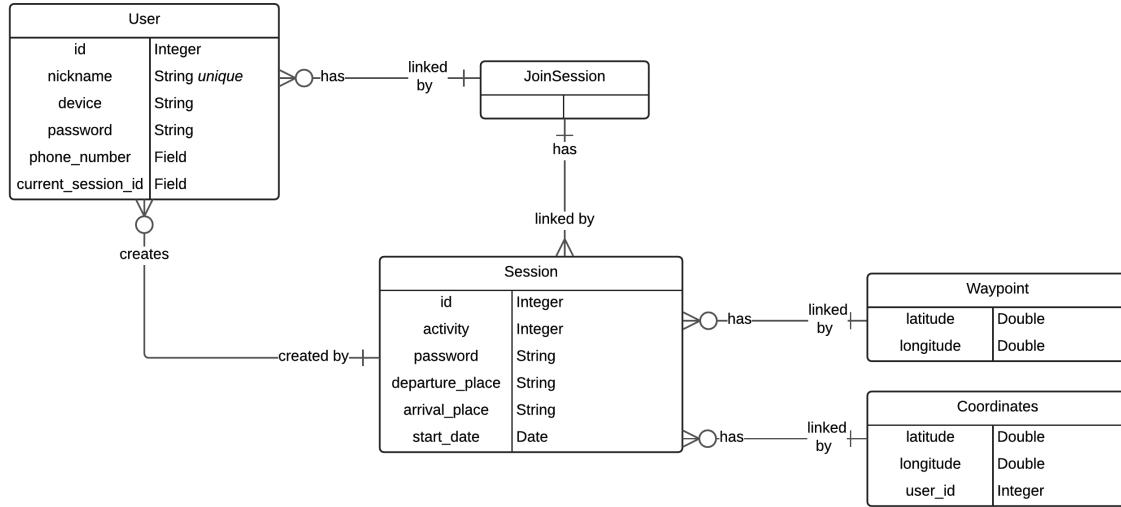
5.1.3.4 Package models

FIGURE 5.6 – Diagramme de classe du package models



5.1.4 Diagramme entité association

FIGURE 5.7 – Diagramme entité association



5.2 Modélisation des axes fonctionnel et dynamique

5.2.1 Généralités

Tous les diagrammes de séquences et d'activités présentés ci-dessous seront depuis le point de vue du client.

Une rapide modélisation du serveur sera présente dans la partie "Modélisation du serveur".

Tous les cas d'utilisation débutent nécessairement par une connexion de l'utilisateur.

De plus l'ensemble des données sont testées sur le client mais aussi sur le serveur qui contient la base de données.

Avant sa première connexion, l'utilisateur devra créer un compte pour obtenir ses informations de connexion. Cette étape est primordiale, car certaines données sont sensibles. Elles seront utilisées directement par l'application comme le pseudonyme ou le numéro de téléphone.

FIGURE 5.8 – Diagramme de séquence authentification

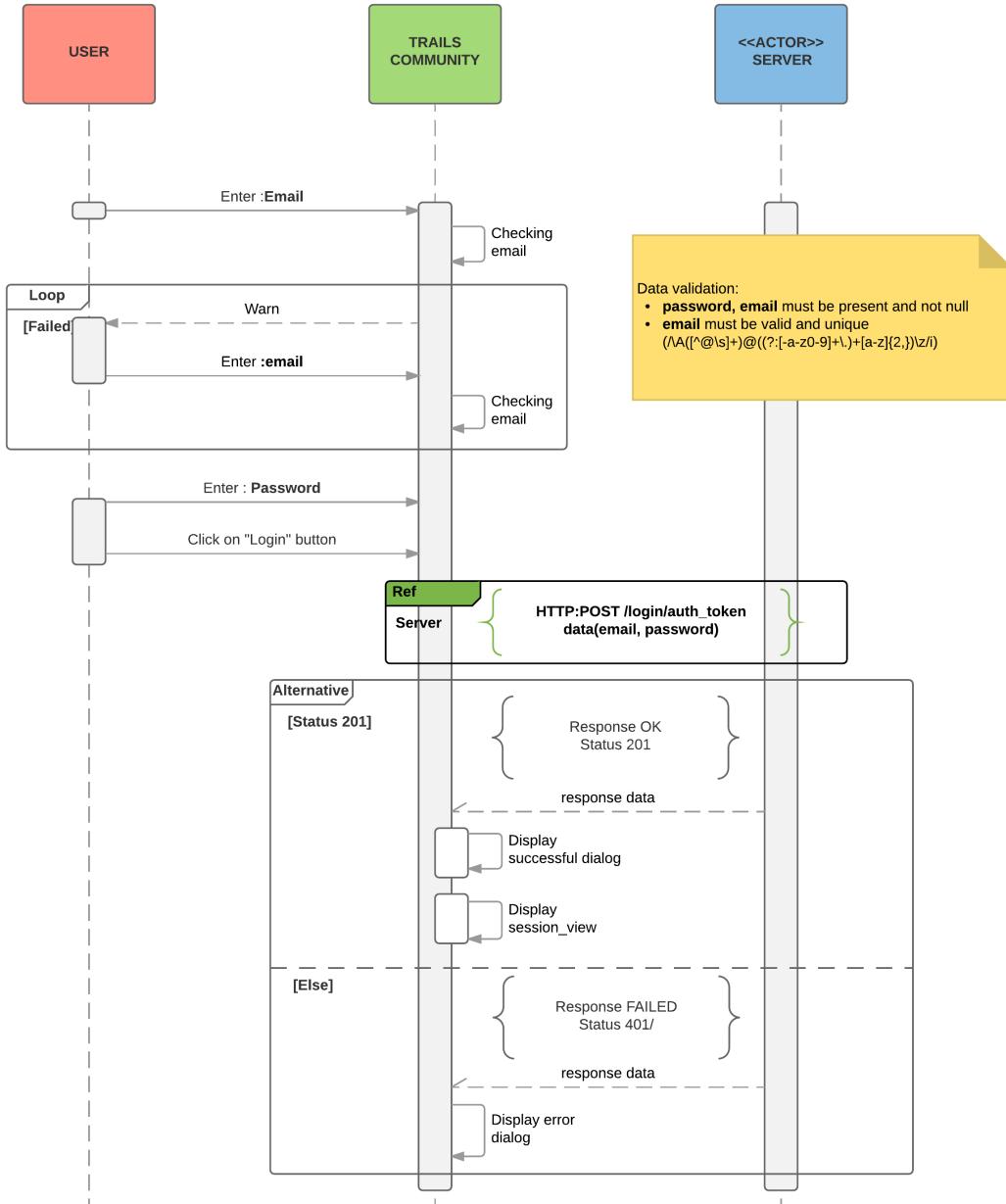
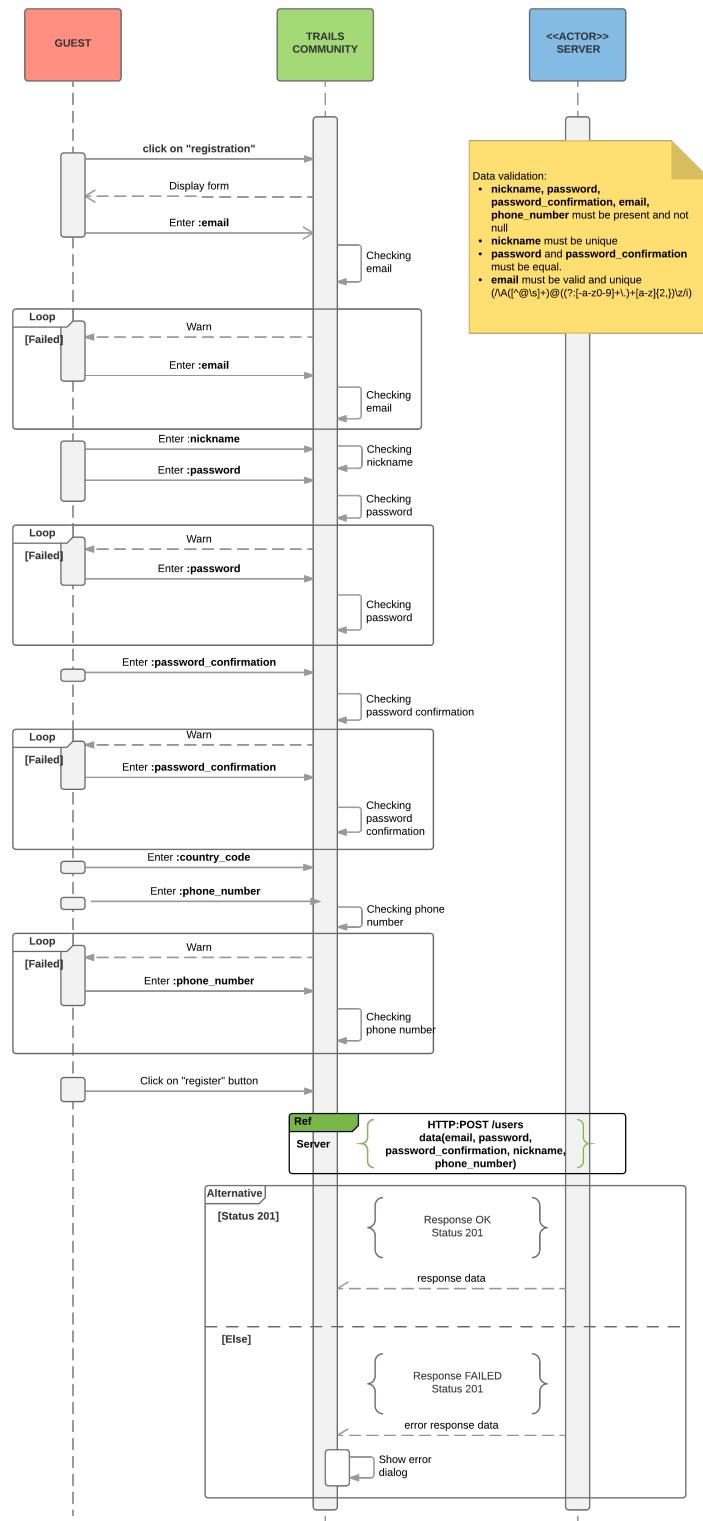


FIGURE 5.9 – Diagramme de séquence créer un compte utilisateur

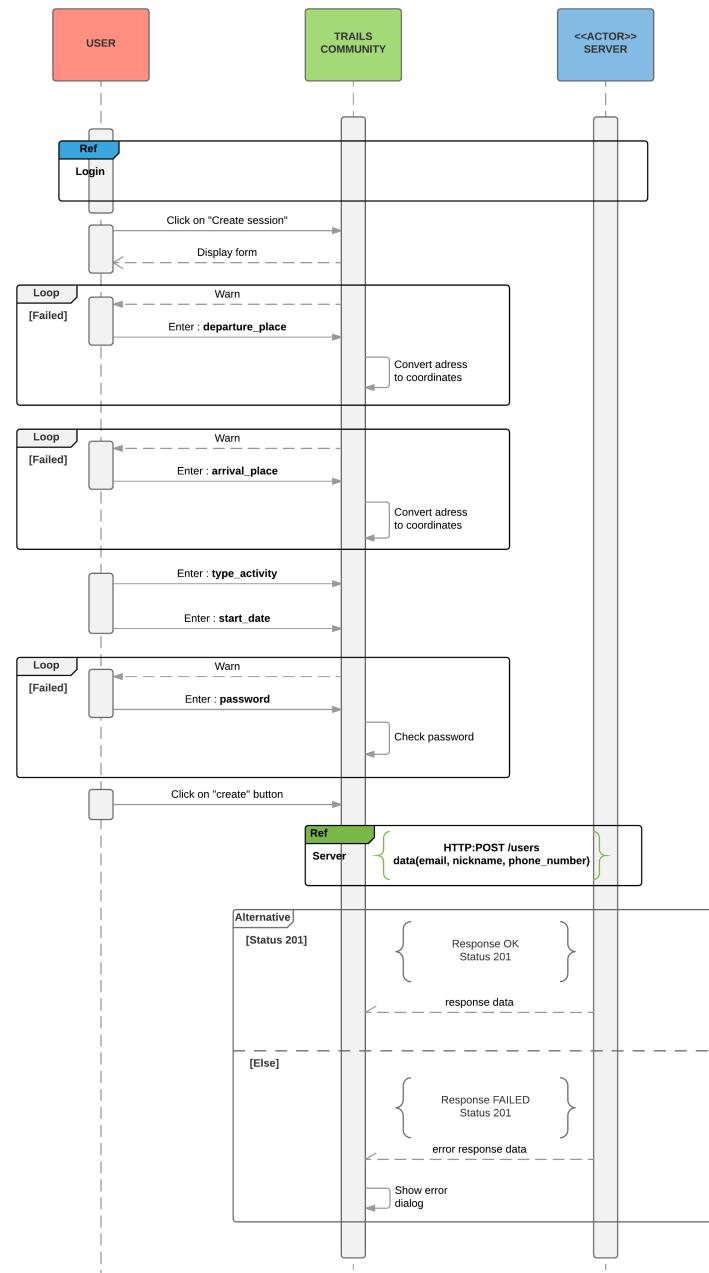


5.2.2 Traitement d'un cas d'utilisation - Créer une session

Le cas d'utilisation que nous avons choisi de détailler ici est la création d'une session. C'est un des cas d'utilisation essentiel au fonctionnement de notre application mobile. Sans cette étape, il est impossible de réaliser n'importe quelles actions concrètes sur l'application.

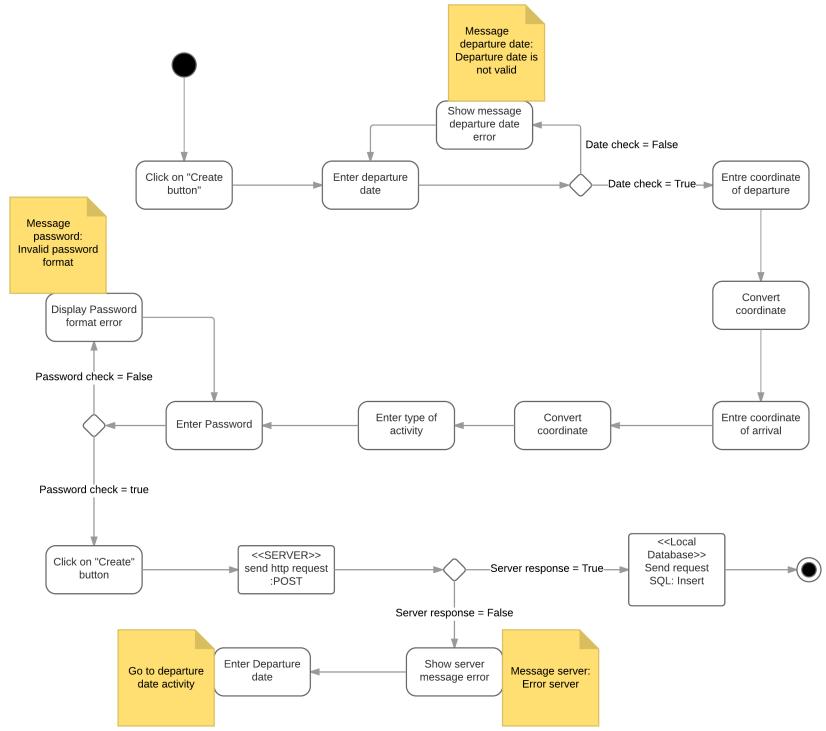
Le diagramme de séquence s'articule principalement autour de trois étapes : l'authentification, la saisie des données du sondage puis l'envoi des données au serveur.

FIGURE 5.10 – Diagramme de séquence créer une session



Comme on peut le voir sur le diagramme d'activité. La gestion d'erreur, ce passe sur les 2 applications : le client et le serveur. De plus, le serveur se charge de remonter les erreurs sur le format des données ou la lecture/écriture en base de données. Le client informe l'utilisateur sur l'échec du format des données entré en paramètre et aussi, l'échec des opération serveur.

FIGURE 5.11 – Diagramme d'activité de la création d'une session



5.2.3 Traitement d'un cas d'utilisation - Rejoindre une session

FIGURE 5.12 – Diagramme de séquence rejoindre une session

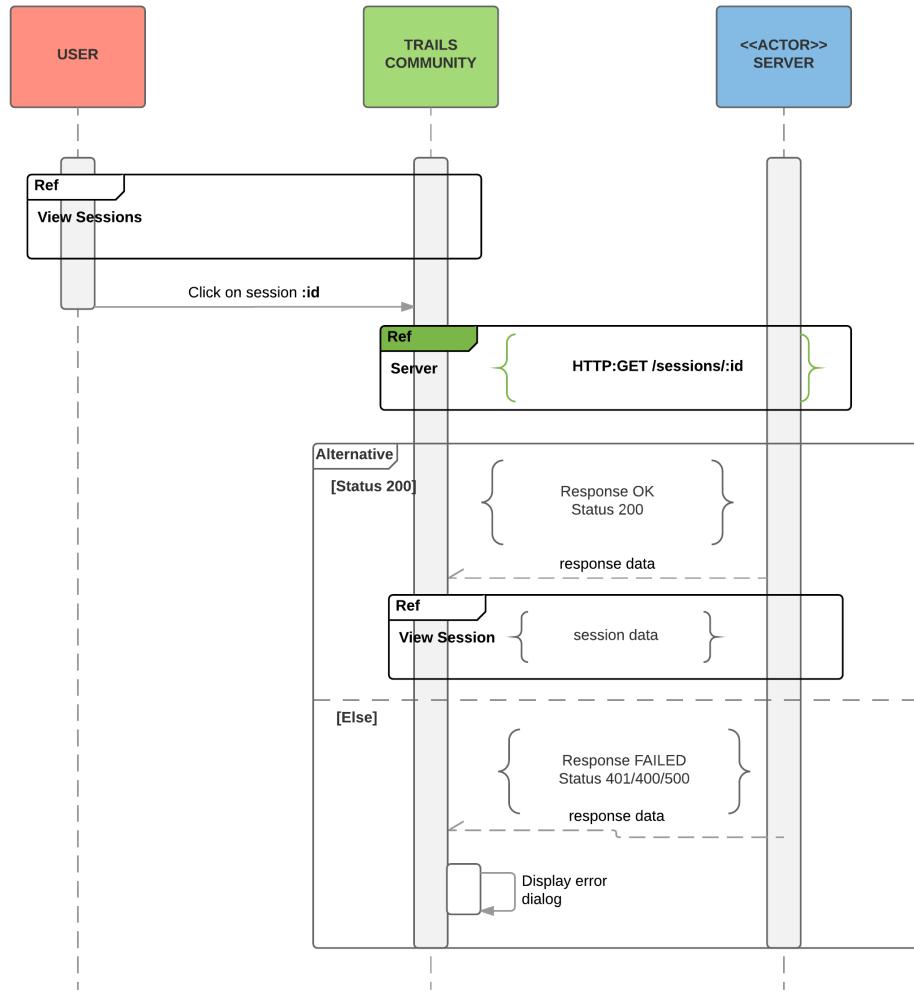


FIGURE 5.13 – Diagramme d'activité rejoindre une session

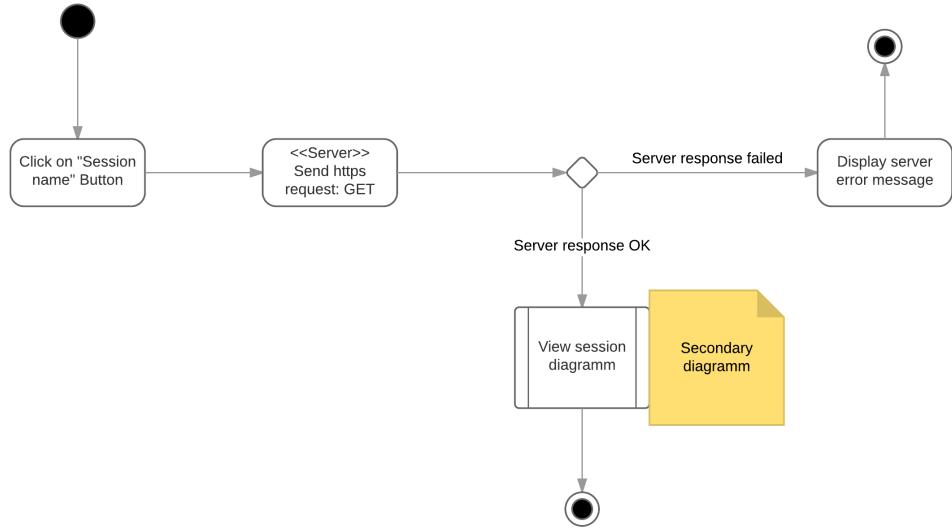


FIGURE 5.14 – Diagramme de séquence de l'affichage d'une session

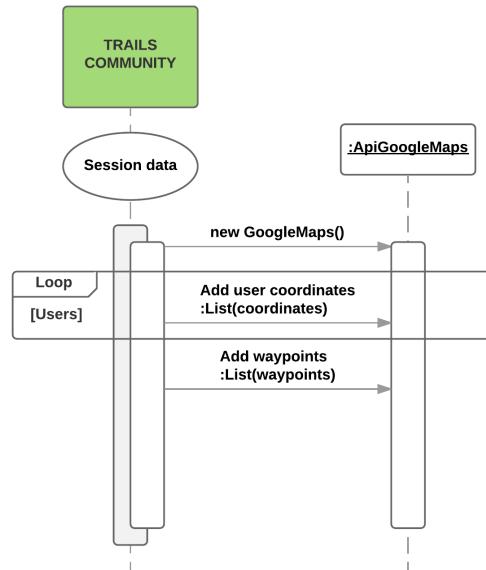
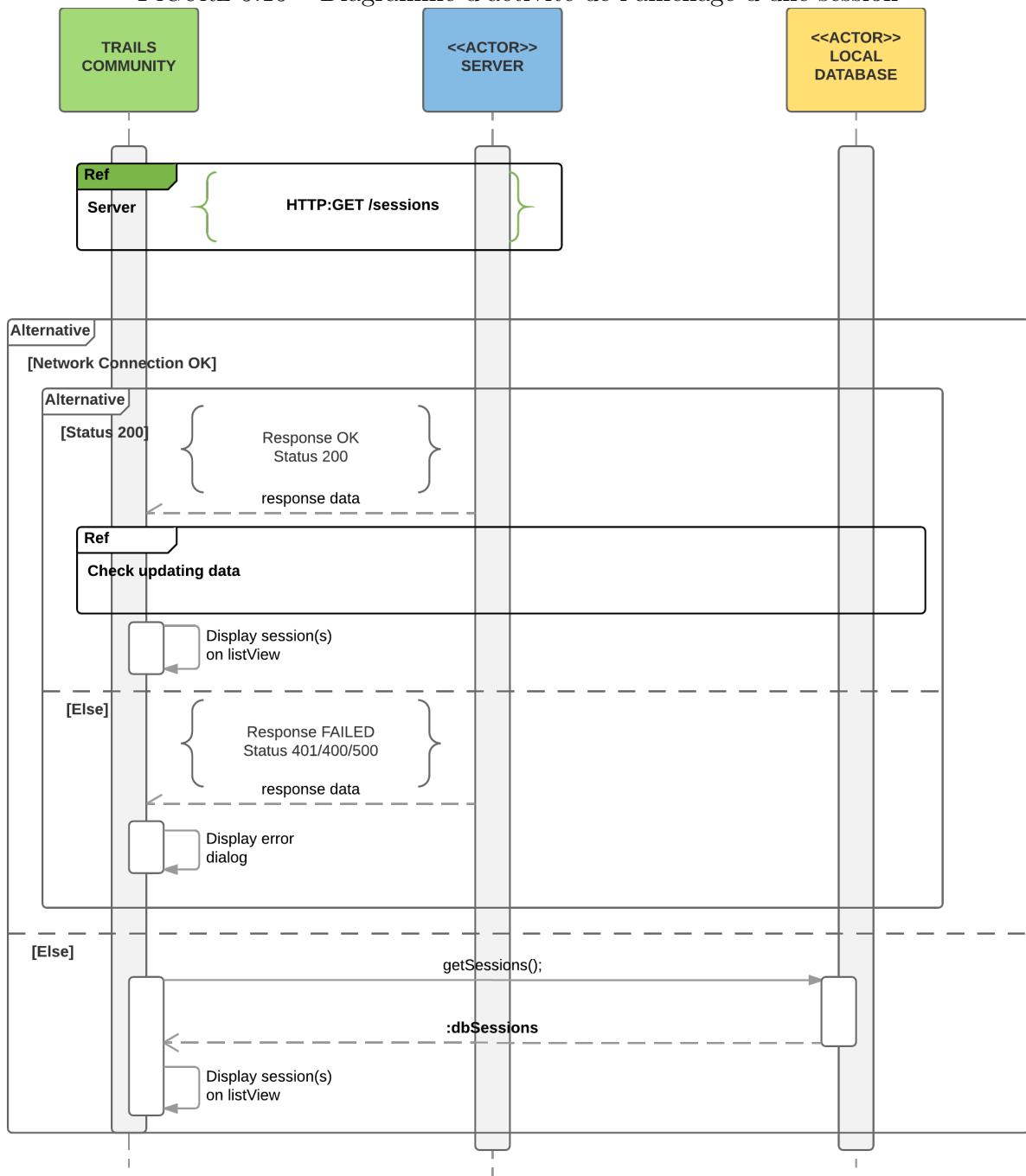


FIGURE 5.15 – Diagramme d’activité de l’affichage d’une session



5.3 Test

L'ensemble des fonctionnalités de l'application seront tester avec un panel de test important. Du côté du client, des tests unitaires ainsi que des tests graphiques seront réalisés pour tester la robustesse de l'application et son bon comportement.

5.3.1 Test unitaire

Identifiant	Titre	Description
1	JSON de la session valide	Teste la conversion d'un JSON en objet Session
2	Parse d'une date invalide	Teste la conversion d'un JSON non-valide, c'est-à-dire qui ne possède pas de date en objet Session
3	JSON session invalide	Teste la conversion du JSON en objet Session
4	Teste la validité d'un email	Teste via une expression régulière si l'e-mail est valide ou non
5	Teste la non-validité d'un email	Teste un tableau d'emails non valide

5.3.2 Test UI

Identifiant	Titre	Description
1	Teste tout les champs de la vue LoginActivity	Entre pour chaque champ une donnée valide pour tester l'authentification d'un utilisateur
2	Teste si le champ e-mail est valide de la vue LoginActivity	Entre un e-mail pour tester si la validation de l'email fonctionne
3	Teste l'affichage d'erreur sur le champ e-mail de la vue LoginActivity	Entre un e-mail invalide pour tester l'affichage de l'erreur
4	Teste le champ mot de passe de la vue LoginActivity	Entre un mot de passe non valide pour tester l'affichage de l'erreur
5	Teste si une erreur s'affiche lorsque le mot de passe est vide dans la vue LoginActivity	Entre un mot de passe vide pour tester l'affichage de l'erreur
6	Teste tous les champs de la vue SingupActivity	Entre pour chaque champ une données valide pour tester la création d'un compte utilisateur
7	Teste l'affichage d'erreur pour le champ mot de passe de la vue SingupActivity	Entre un mot de passe vide pour tester l'affichage d'erreur du mot de passe vide
8	Teste si la forme de l'e-mail est valide de la vue SingupActivity	Entre un e-mail invalide pour tester l'affichage d'erreur de l'e-mail invalide
9	Teste si les 2 mots de passe sont valides de la vue SingupActivity	Entre 2 mots de passe différent pour tester l'affichage d'erreur lorsque les mots de passe sont différents
10	Teste le cas d'erreur d'un mot de passe vide de la vue SingupActivity	Entre un mot de passe vide pour afficher le cas d'erreur
11	Teste le cas d'erreur d'un pseudonyme vide de la vue SingupActivity	Entre un champs vide pour le pseudonyme pour tester l'affichage du cas d'erreur
12	Teste le cas d'erreur d'un numéro de téléphone vide de la vue SingupActivity	Entre un champs vide pour le numéro de téléphone pour tester l'affichage du cas d'erreur
13	Teste l'ajout d'un waypoint de la vue SessionActivity	Ajoute un waypoint sur la Google maps
14	Teste l'action du bouton chat de la vue SessionActivity	Teste si le chat s'affiche bien lors du clique de l'utilisateur sur le bouton chat

15	Teste l'ajout d'un message dans le chat de la vue SessionActivity	Ajoute un message dans le chat pour tester son affichage et son ajout dans la liste du chat
16	Teste le cas d'erreur dans un message vide dans le chat de la vue SessionActivity	Ajoute un message vide dans le chat pour tester l'affichage du cas d'erreur
17	Teste la fermeture du chat de la vue SessionActivity	Teste si le chat se ferme bien lors du clique de l'utilisateur sur le bouton chat
18	Teste la conversion d'une adresse en coordonnées GPS dans la vue SessionFormActivity	Teste la conversion d'une adresse en coordonnées GPS
19	Teste la création d'une session de la vue SessionFormActivity	Ajout dans chaque champ de la vue une entrée valide pour tester la création d'une session
20	Teste l'affichage d'erreur du champs lieu de départ de la vue SessionFormActivity	Teste pour le champ lieu de départ si l'affichage du cas d'erreur fonctionne si le champ est vide
21	Teste l'affichage d'erreur du champs lieu de d'arrivée de la vue SessionFormActivity	Teste pour le champ lieu de d'arrivé si l'affichage du cas d'erreur fonctionne si le champ est vide
22	Teste l'affichage d'erreur du champs date de départ de la vue SessionFormActivity	Teste pour le champ date de départ si l'affichage du cas d'erreur fonctionne si le champ est vide
23	Teste l'affichage d'erreur du champs mot de passe de la vue SessionFormActivity	Teste pour le champ mot de passe si l'affichage du cas d'erreur fonctionne si le champ est vide
24	Teste la modification des données de l'utilisateur de la vue ProfileActivity	Ajoute dans tous les champs des données valide pour tester la modification des données de l'utilisateur courant
25	Teste l'affichage d'erreur du champs pseudonyme de la vue ProfileActivity	Teste pour le champ pseudonyme si l'affichage du cas d'erreur fonctionne si le champ est vide
26	Teste l'affichage d'erreur du champs numéro de téléphone de la vue ProfileActivity	Teste pour le champ numéro de téléphone si l'affichage du cas d'erreur fonctionne si le champ est vide

Deuxième partie

Serveur TrailsCommunity

Chapitre 1

Introduction

1.1 Conventions

- **Client** : Application client.
- **Statut** : Code réponse de la requête HTTP
- Toutes les réponses possibles sont listés sous la catégorie 'Réponse'
- Toutes les réponses sont du format JSON

1.2 Codes statut

Tous les codes de statut sont des standards HTTP. Cette API utilisera les suivants :

- $2XX$ - Succès de la requête
- $4XX$ - Erreur du côté client
- $5XX$ - Erreur du côté serveur

Code statut	Description
200	Requête traitée avec succès
201	Requête traitée avec succès et création d'un document
202	Requête traitée avec succès mais pas d'information à renvoyer
400	La syntaxe de la requête est erronée
401	Une authentification est nécessaire pour accéder à la ressource
404	Ressource non trouvée
500	Erreur interne du serveur
503	Service temporairement indisponible ou en maintenance

Chapitre 2

Ressources

2.1 Authentification

2.1.1 Requête

Méthode	URL
<i>POST</i>	api/login/auth_token

Type	Paramètres	Valeurs
<i>HEADER</i>	Application/json	<i>String</i>
<i>JSON</i>	{ "auth": { "email": "mail@mail.com", "password": "FZ,f235nDE", } }	<i>String</i> <i>String</i>

email : correspond à l'adresse mail valide de l'utilisateur

password : mot de passe de l'utilisateur

2.1.2 Réponse

Statut	Réponse
200	{ "jwt": <Authorization> }
401	
500	

Authorization(string) - Tous les autres appels nécessitant une authentification doivent avoir cette clé dans le header.

2.2 Inscription

2.2.1 Requête

Enregistrement d'un utilisateur dans le système.

Méthode	URL
<i>POST</i>	api/users/

Type	Paramètres	Valeurs
<i>HEADER</i>	Application/json	<i>String</i>
<i>JSON</i>	{ "nickname": "Arthur83", "email": "apaul@custom.com", "phone_number": "+33651678908", "password": "myPassword33", "password_confirmation": "myPassword33" }	<i>String</i> <i>String</i> <i>String</i> <i>String</i> <i>String</i>

nickname : pseudonyme de l'utilisateur

email : email de l'utilisateur non nul et valide

password : mot de passe d'utilisateur

password_confirmation : confirmation du mot de passe de l'utilisateur. Il doit être identique à *password*.

2.2.2 Réponse

Statut	Réponse
201	<pre>{ "data": { "id": 2, "nickname": "Arthur83", "phone_number": "+33651678908", "current_session_id": null } }</pre>
400	<pre>{ "errors": { "nickname": ["has already been taken"], "email": ["is invalid"] } }</pre>
500	

2.3 Ajout d'une session

2.3.1 Requête

Permet d'ajouter une nouvelle session dans le serveur. Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
<i>POST</i>	<i>api/sessions/</i>

Type	Paramètres	Valeurs
<i>HEADER</i>	Application/json	<i>String</i>
<i>HEADER</i>	Authorization	<i>String</i>
<i>JSON</i>	<pre>{ "activity": 1, "password": "lockSessionPsw", "departure_place": "43.179363;5.717782", "arrival_place": "43.191168;5.730819", "start_date": "2016-11-20" }</pre>	<i>Integer</i> <i>String</i> <i>String</i> <i>String</i> <i>Date</i>

activity : identifiant de l'activité. (1 : randonnée, 2 : vélo, ...)

password : mot de passe de la session, permet de restreindre l'entrée.

departure_place : lieu de départ. Couple de position géographique (*latitude*; *longitude*)

arrival_place : lieu d'arrivée. Couple de position géographique (*latitude*; *longitude*)

2.3.2 Réponse

Statut	Réponse
200	<pre>{ "data": { "id": 5, "activity": 1, "departure_place": "43.179363;5.717782", "arrival_place": "43.191168;5.730819", "start_date": "2016-11-20", "close": false, "lock": true, "user": { "id": 1, "nickname": "Raptor234", "phone_number": "+33623569450", "current_session_id": null }, "coordinates": [], "waypoints": [] } }</pre>
400	<pre>{ "errors": { "activity": ["can't be blank", "is not a number"] } }</pre>
401	
500	

2.4 Liste des sessions

2.4.1 Requête

Permet de récupérer la liste des sessions du système. Cette ressource est contrainte d'un token d'authentification <*Authorization*> qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
<i>GET</i>	api/sessions/

Type	Paramètres	Valeurs
<i>HEADER</i>	Authorization	<i>String</i>

2.4.2 Réponse

Statut	Réponse
200	<pre>{ "data": [{ "id": 1, "activity": 1, "departure_place": "43.179363;5.717782", "arrival_place": "43.191168;5.730819", "start_date": "2016-11-20", "close": false, "lock": true, "user": { "id": 2, "nickname": "Arthur83", "phone_number": "+33651678908", "current_session_id": null } }, { "id": 2, "activity": 3, "departure_place": "43.179363;5.717782", "arrival_place": "43.191168;5.730819", "start_date": "2016-11-20", "close": false, "lock": true, "user": { "id": 2, "nickname": "Arthur83", "phone_number": "+33651678908", "current_session_id": null } }] }</pre>
401	
500	

2.5 Session

2.5.1 Requête

Permet de récupérer une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
<i>GET</i>	api/sessions/:id

Type	Paramètres	Valeurs
<i>URL_PARAM</i>	id	<i>Integer</i>
<i>HEADER</i>	Authorization	<i>String</i>

id : identifiant de la session voulu

2.5.2 Réponse

Statut	Réponse
200	<pre>{ "data": { "id": 2, "activity": 3, "departure_place": "43.179363;5.717782", "arrival_place": "43.191168;5.730819", "start_date": "2016-11-20", "close": false, "lock": true, "user": { "id": 2, "nickname": "Arthur83", "phone_number": "+33651678908", "current_session_id": null }, "coordinates": [{ "latitude": 43.179363, "longitude": 5.717782, "user_id": 1 }], "waypoints": [{ "latitude": 43.179363, "longitude": 5.717782 }] } }</pre>
401	
404	<pre>{ "errors": { "session": "Record Not Found" } }</pre>
500	

2.6 Mis à jour d'une session

2.6.1 Requête

Permet de mettre à jour session spécifique selon un identifiant (ID) dans le système. Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
<i>PUT</i>	api/sessions/:id

Type	Paramètres	Valeurs
<i>HEADER</i>	Application/json	<i>String</i>
<i>HEADER</i>	Authorization	<i>String</i>
<i>JSON</i>	{ "close": true }	<i>boolean</i>

close : boolean déterminant si la session doit être fermée.

2.6.2 Réponse

Statut	Réponse
200	{ "data": { "id": 7, "activity": 3, "departure_place": "43.179363;5.717782", "arrival_place": "43.191168;5.730819", "start_date": "2016-11-20", "close": true, "lock": true, "user": { "id": 2, "nickname": "Arthur83", "phone_number": "+33651678908", "current_session_id": 1 }, "coordinates": [], "waypoints": [] } }
401	
500	

2.7 Rejoindre une session

2.7.1 Requête

Permet de à l'utilisateur de rejoindre une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*. De plus, un mot de passe peut être demandé si la session est restreinte.

Méthode	URL
<i>GET</i>	api/sessions/:id/join

Type	Paramètres	Valeurs
<i>HEADER</i>	Authorization	<i>String</i>
<i>URL_PARAM</i>	id	<i>Integer</i>
<i>URL_PARAM</i>	password (optional)	<i>String</i>

id : identifiant de la session voulu

2.7.2 Réponse

Statut	Réponse
200	
400	{ "errors": { "session": "Bad password." OR "session": "This session needs a password to join it." } }
401	
404	{ "errors": { "session": "Record Not Found" } }
500	

2.8 Ajout d'un waypoint

2.8.1 Requête

Permet de à l'utilisateur d'ajouter une nouveau waypoint pour une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
<i>POST</i>	api/sessions/:id/waypoint

Type	Paramètres	Valeurs
<i>URL_PARAM</i>	id	<i>Integer</i>
<i>HEADER</i>	Authorization	<i>String</i>
<i>JSON</i>	{ "latitude": 43.179363, "longitude": 5.717782 }	<i>Double</i> <i>Double</i>

id : identifiant de la session voulu

2.8.2 Réponse

Statut	Réponse
200	
401	
404	{ "errors": { "session": "Record Not Found" } }
500	

2.9 Ajout d'une coordonnée

2.9.1 Requête

Permet de à l'utilisateur d'ajouter une nouvelle coordonnée pour une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
<i>POST</i>	api/sessions/:id/waypoint

Type	Paramètres	Valeurs
<i>URL_PARAM</i>	id	<i>Integer</i>
<i>HEADER</i>	Authorization	<i>String</i>
<i>JSON</i>	{ "latitude": 43.179363, "longitude": 5.717782 }	<i>Double</i> <i>Double</i>

id : identifiant de la session voulu

2.9.2 Réponse

Statut	Réponse
200	
401	
404	{ "errors": { "session": "Record Not Found" } }
500	

2.10 Envoyer un message

2.10.1 Requête

Permet de à l'utilisateur d'envoyer une nouveau message pour une session spécifique selon son identifiant (ID). Cette ressource est contrainte d'un token d'authentification *<Authorization>* qui a été donnée avec la ressource *api/login/auth_token*

Méthode	URL
<i>POST</i>	api/chats/:id

Type	Paramètres	Valeurs
<i>URL_PARAM</i>	<i>id</i>	<i>Integer</i>
<i>HEADER</i>	<i>Authorization</i>	<i>String</i>
<i>JSON</i>	{ "message": "mon message ... " }	<i>String</i>

id : identifiant de la session voulu

2.10.2 Réponse

Statut	Réponse
200	
401	
404	{ "errors": { "session": "Record Not Found" } }
500	

2.11 Test

2.11.1 Test unitaire

Identifiant	Titre	Description
SER_UT_1	Modèle session	Teste la validité du modèle Session
SER_UT_2	Modèle utilisateur	Teste la validité du modèle User
SER_UT_3	Modèle coordonnées	Teste la validité du modèle Coordinate
SER_UT_4	Modèle waypoint	Teste la validité du modèle Waypoint
SER_UT_5	Modèle message	Teste la validité du modèle Chat
SER_UT_6	Création d'une session	Teste la la création d'une session
SER_UT_7	Création d'une session	Teste si l'utilisateur est autorisé
SER_UT_8	Création d'une session	Teste avec de mauvais paramètres
SER_UT_9	Récupération d'une session	Teste la liste de sessions reçue
SER_UT_10	Récupération d'une session	Teste si l'utilisateur est autorisé
SER_UT_11	Mise à jour d'une session	Teste la la mise à jour d'une session
SER_UT_12	Mise à jour d'une session	Teste si l'utilisateur est autorisé
SER_UT_13	Mise à jour d'une session	Teste avec de mauvais paramètres
SER_UT_14	Suppression d'une session	Teste de la suppression
SER_UT_15	Suppression d'une session	Teste si l'utilisateur est autorisé
SER_UT_16	Creation d'un utilisateur	Teste la création d'une utilisateur
SER_UT_17	Creation d'un utilisateur	Teste avec de mauvais paramètres

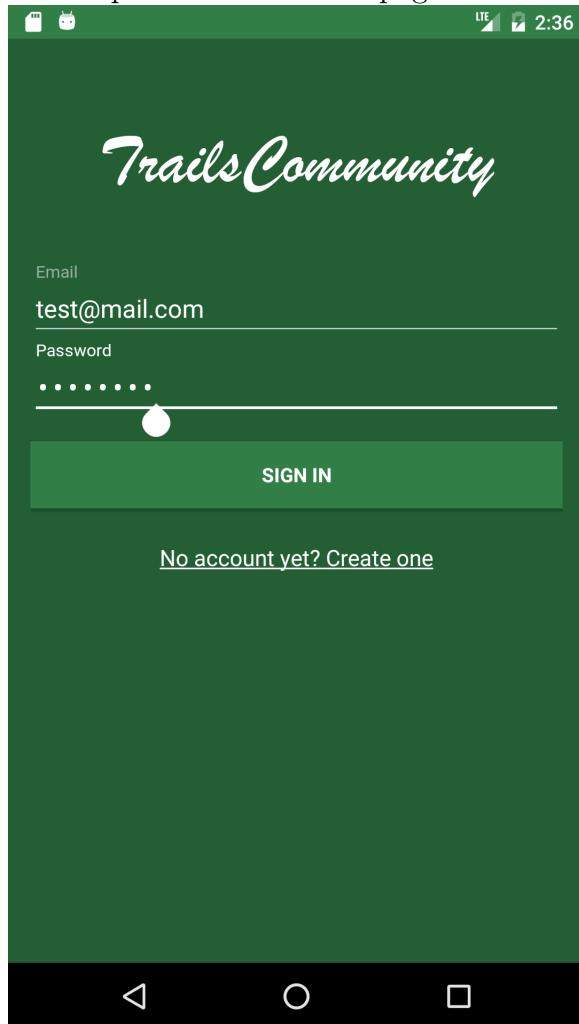
Identifiant	Titre	Description
SER_UT_18	Creation d'une coordonnée	Teste la création d'une coordonnée
SER_UT_19	Creation d'une coordonnée	Teste avec de mauvais paramètres
SER_UT_20	Creation d'une coordonnée	Teste si l'utilisateur est autorisé
SER_UT_21	Creation d'un waypoint	Teste la création d'un waypoint
SER_UT_22	Creation d'un waypoint	Teste avec de mauvais paramètres
SER_UT_23	Creation d'un waypoint	Teste si l'utilisateur est autorisé
SER_UT_24	Creation d'un message	Teste la création d'un message
SER_UT_25	Creation d'un message	Teste avec de mauvais paramètres
SER_UT_26	Creation d'un message	Teste si l'utilisateur est autorisé
SER_UT_27	Rejoindre une session	Teste avec de mauvais paramètres
SER_UT_28	Rejoindre une session	Teste si l'utilisateur est autorisé
SER_UT_29	Requête HTTP	Teste des codes retour
SER_UT_30	Rejoindre une session	Teste si l'utilisateur est autorisé

Troisième partie

Manuel utilisateur

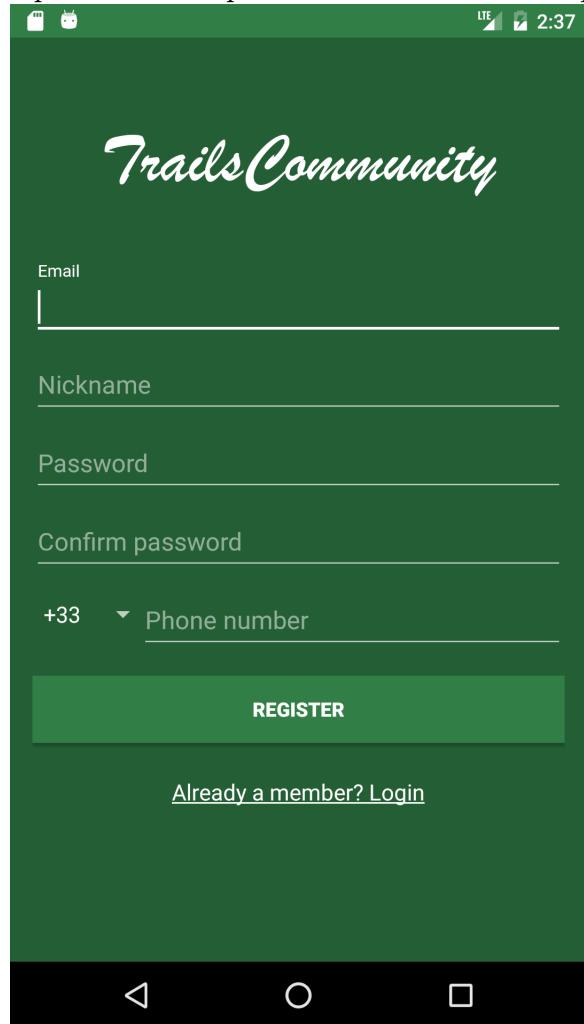
Il est nécessaire de s'authentifier via un compte utilisateur précédemment créé pour pouvoir utiliser l'ensemble des fonctionnalités de l'application TrailsCommunity.

FIGURE 1 – Capture d'écran de la page d'authentification



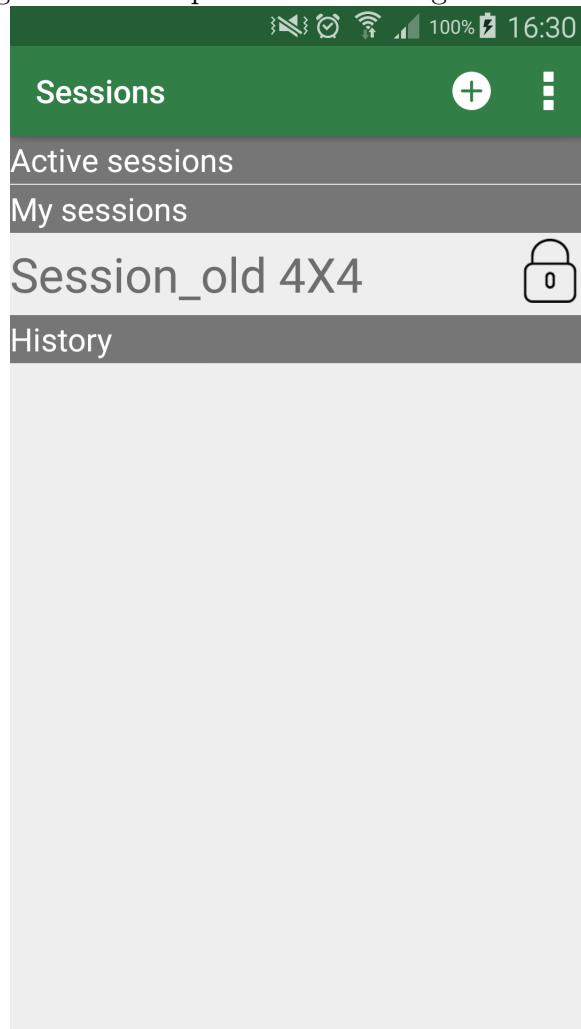
Si l'utilisateur ne possède pas un compte, il est possible d'en créer un sur l'application. Différentes informations lui seront demandées.

FIGURE 2 – Capture d'écran pour la création d'un compte utilisateur



Une fois connecté, plusieurs choix s'offre à lui. Il peut alors sélectionner une session active pour la rejoindre ou visualiser son historique. Le bouton plus sur la barre de menu lui permet de créer une nouvelle session. D'autres fonctionnalités sont possibles en cliquant sur le petit menu en haut à gauche.

FIGURE 3 – Diagramme de séquence de l'affichage de l'ensemble des sessions



Pour la création d'une session, différentes informations cruciales sont demandées à l'utilisateur pour le bon déroulement de la futur session. De plus, la modification des données de l'utilisateur courante ainsi que la déconnexion sont disponibles depuis la sélection des sessions.

FIGURE 4 – Capture d'écran de l'affichage de la création d'une session

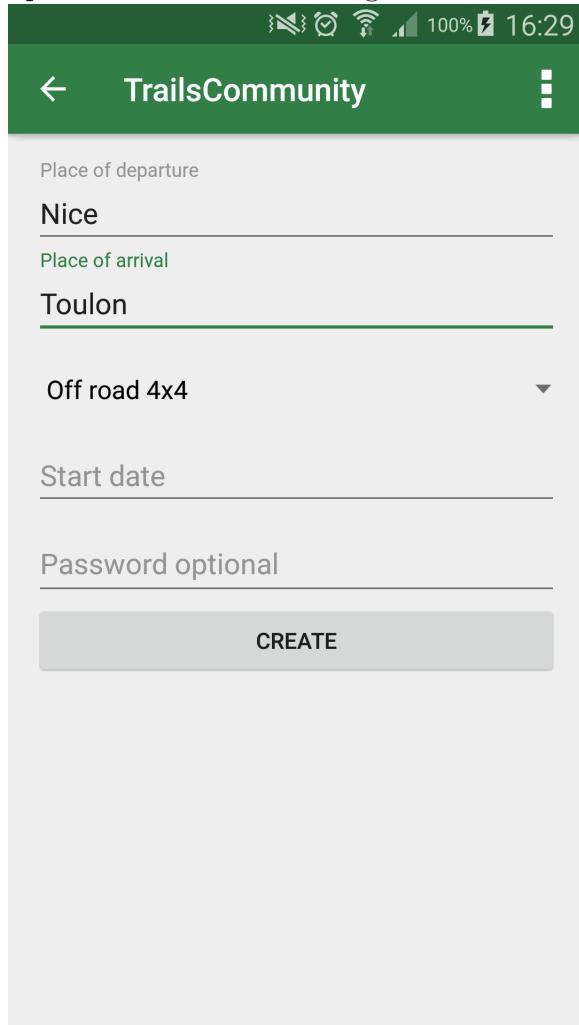
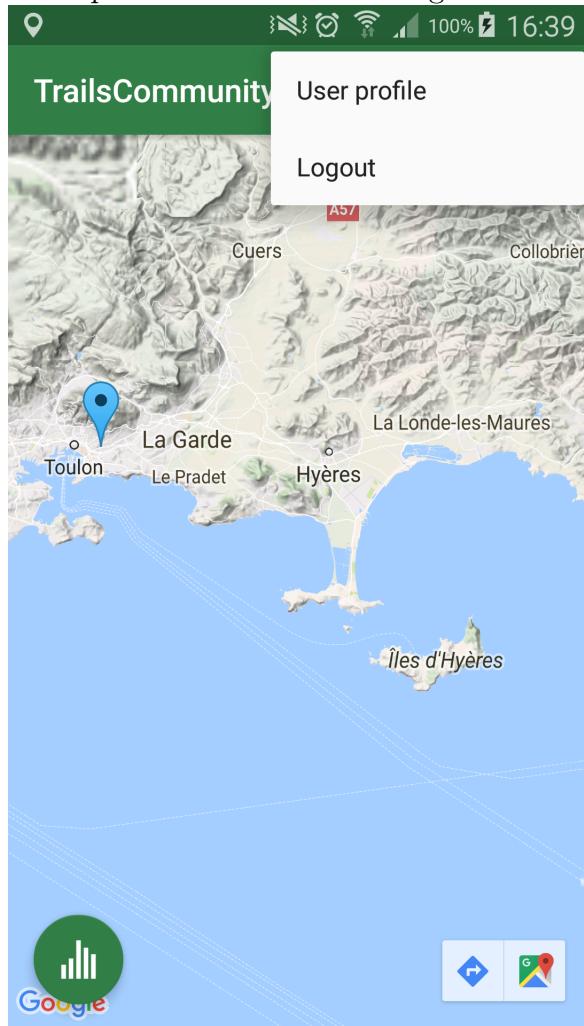
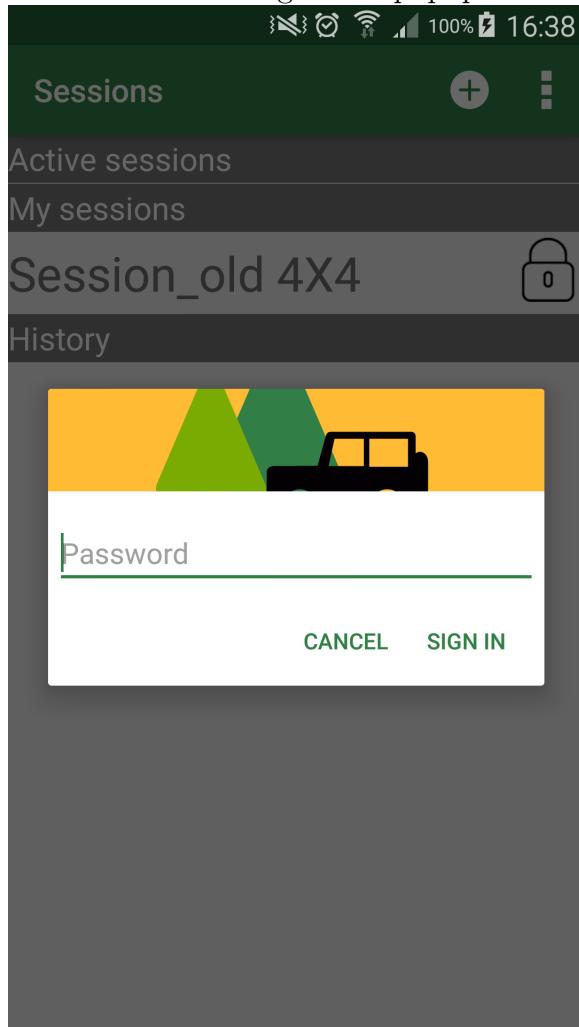


FIGURE 5 – Capture d'écran de l'affichage de la déconnexion



Lors de la sélection d'une session, si la session est privé, l'application va alors demander à l'utilisateur d'entrer le mot de passe de la session.

FIGURE 6 – Capture d'écran de l'affichage de la popup de l'entrée du mot de passe



Une fois le mot de passe correct, une Google maps s'affiche avec l'ensemble des tracés des différents participants. Différents boutons sont disponibles dans la barre de menu pour afficher le chat ou pour modifier les données de l'utilisateur courant. De plus, les statistiques sont disponibles lorsque l'utilisateur clique sur le bouton flottant en bas à gauche.

FIGURE 7 – Capture d'écran de l'affichage d'une session

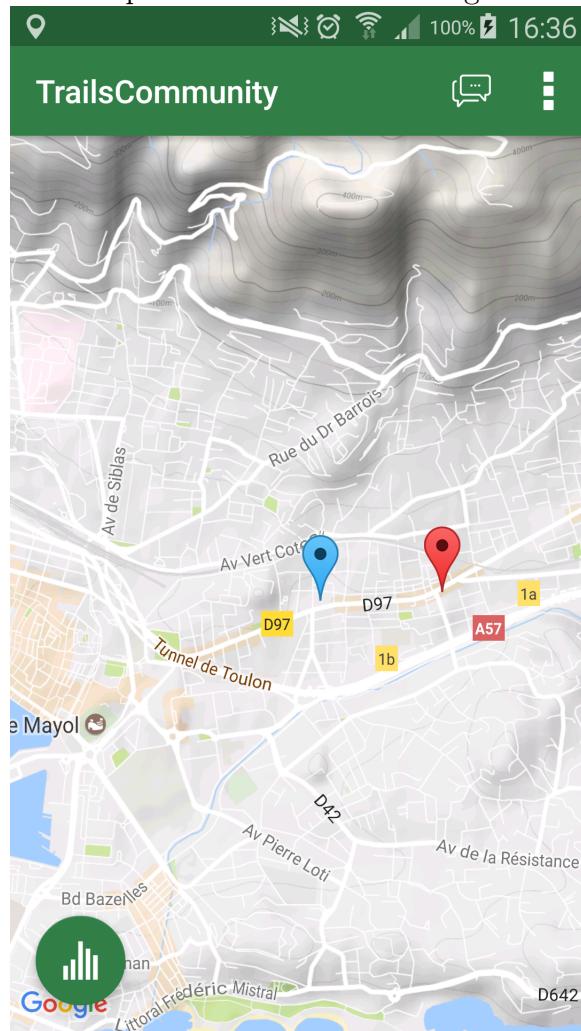


FIGURE 8 – Capture d'écran de l'affichage du chat

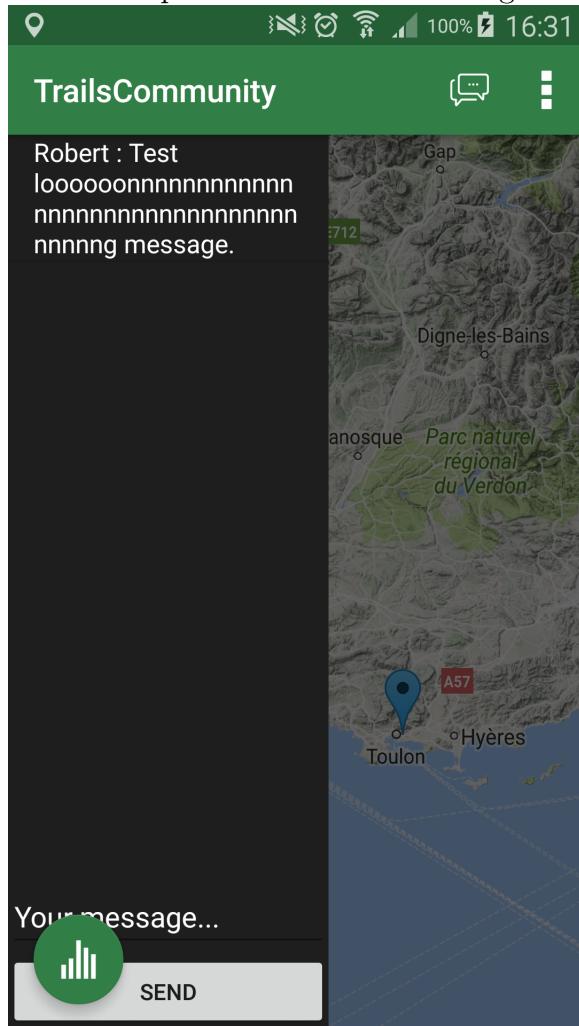
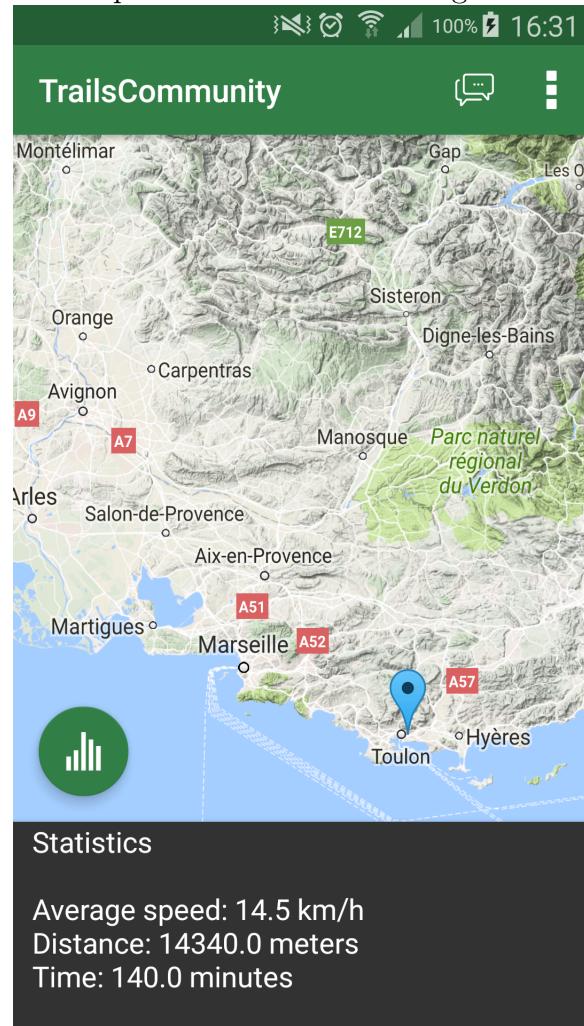


FIGURE 9 – Capture d'écran de l'affichage des statistiques



Quatrième partie

Déroulement du projet

0.1 Difficultés rencontrées

La principale difficulté que nous avons eu lors de la réalisation de ce projet a été de maintenir une cohérence entre les diagrammes UML réalisés lors de la première partie du projet et le code Java réalisé par la suite. Cependant, l'expérience engrangée lors de la réalisation de notre premier projet UML nous a permis d'obtenir un certain recul pour éviter les différents problèmes rencontrés précédemment. Il a donc été nécessaire de reprendre et de mettre à jour la plupart des diagrammes UML après avoir commencé à mettre en place l'application Android.

La seconde partie de difficultés est liée à l'utilisation de nouvelles technologies :

- mise en place d'un serveur afin d'obtenir un service utilisant une architecture propre à notre application mobile : utilisation de Ruby.
- utilisation du système d'annotation d'Android permettant une meilleure lisibilité du code d'une part et donc une meilleur maintenance.
- de nombreuses dépendances et bibliothèques ont été ajoutées dans le projet Android.

Globalement, l'avancée du projet a été assez fluide malgré ces difficultés.

0.2 Perspectives d'évolution

L'application mobile possède les fonctionnalités qui permettent de rejoindre une session et d'y participer d'une manière tout à fait normal, cependant de nombreuses améliorations peuvent lui être apportées comme :

- La modification des données des sessions et des utilisateurs.
- La création d'un site web permettant aux organisateurs de gérer et de collecter l'ensemble des informations des sessions en cours de réalisation.

0.3 Conclusion

La conduite de ce projet nous aura permis de confirmer nos compétences UML utilisées lors du projet précédent. De nombreux problèmes ont pu donc être résolus plus rapidement ou tout simplement évités. De plus, l'ensemble de nos compétences en Java ont pu être appliquées sur un projet Android concret englobant l'ensemble des compétences vues lors du module prévu à cet effet. La réalisation de ce projet, en plus de nous avoir aidés à utiliser de nouveaux outils et de nouvelles connaissances, nous a également permis de conduire un projet plus ambitieux que prévu depuis la phase de conception jusqu'à l'implémentation, dans un délai relativement court.

Cinquième partie

Annexe - Diagrammes UML

Chapitre 1

Diagrammes généraux

FIGURE 1.1 – Diagramme d’architecture de l’API

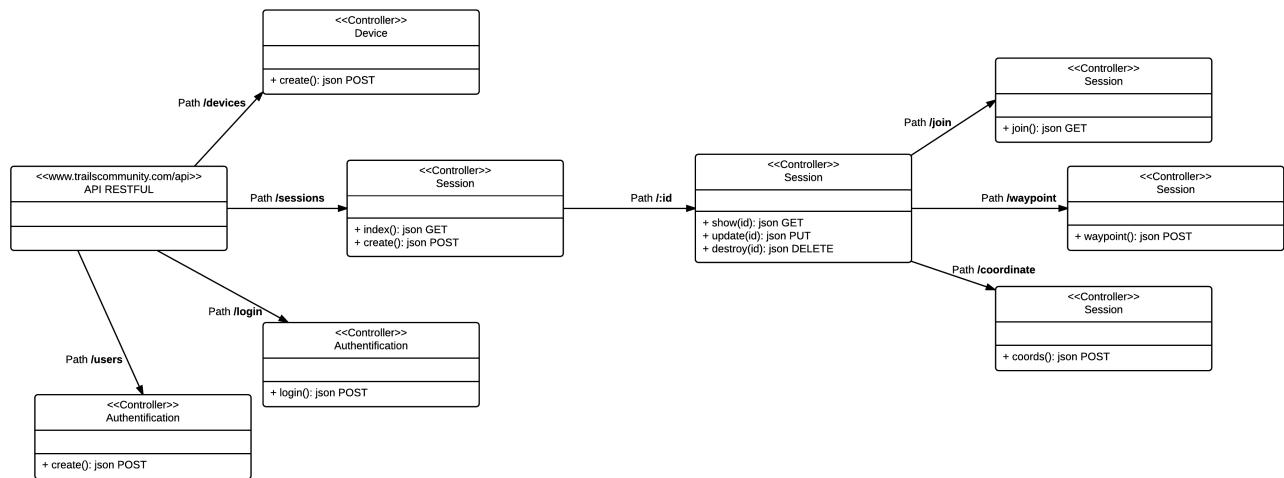
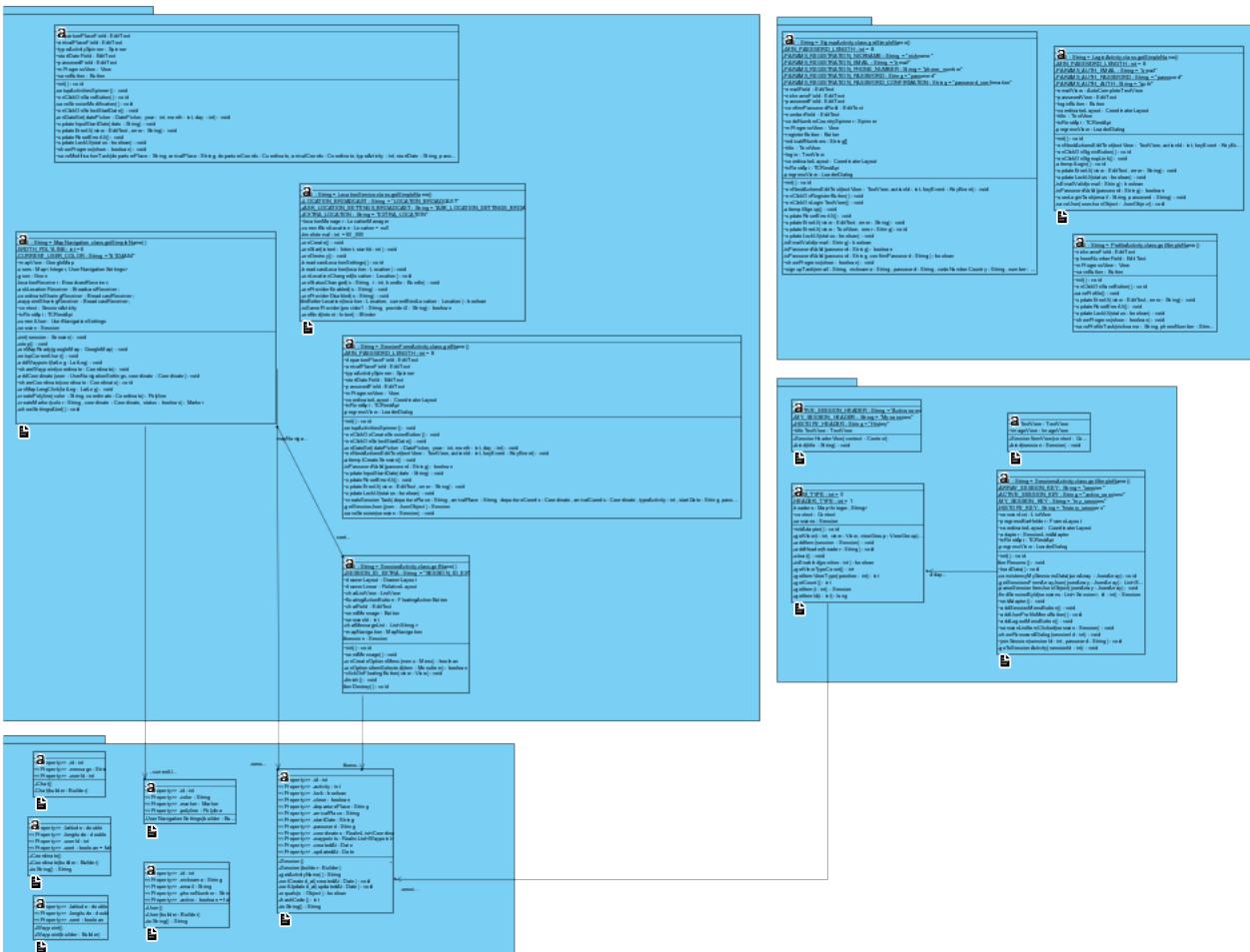


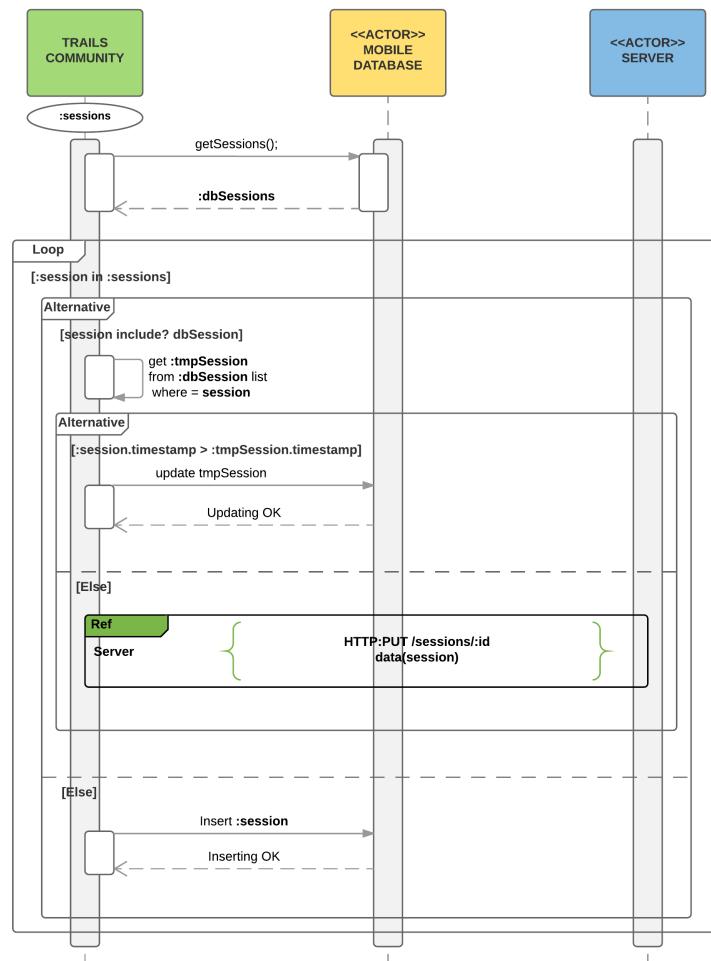
FIGURE 1.2 – Diagramme de classe



Chapitre 2

Modélisation serveur

FIGURE 2.1 – Diagramme de séquence pour la vérification de la cohérence des données entre le client et le serveur



Chapitre 3

Diagrammes de séquences

FIGURE 3.1 – Diagramme de séquence de l'ajout d'un waypoint

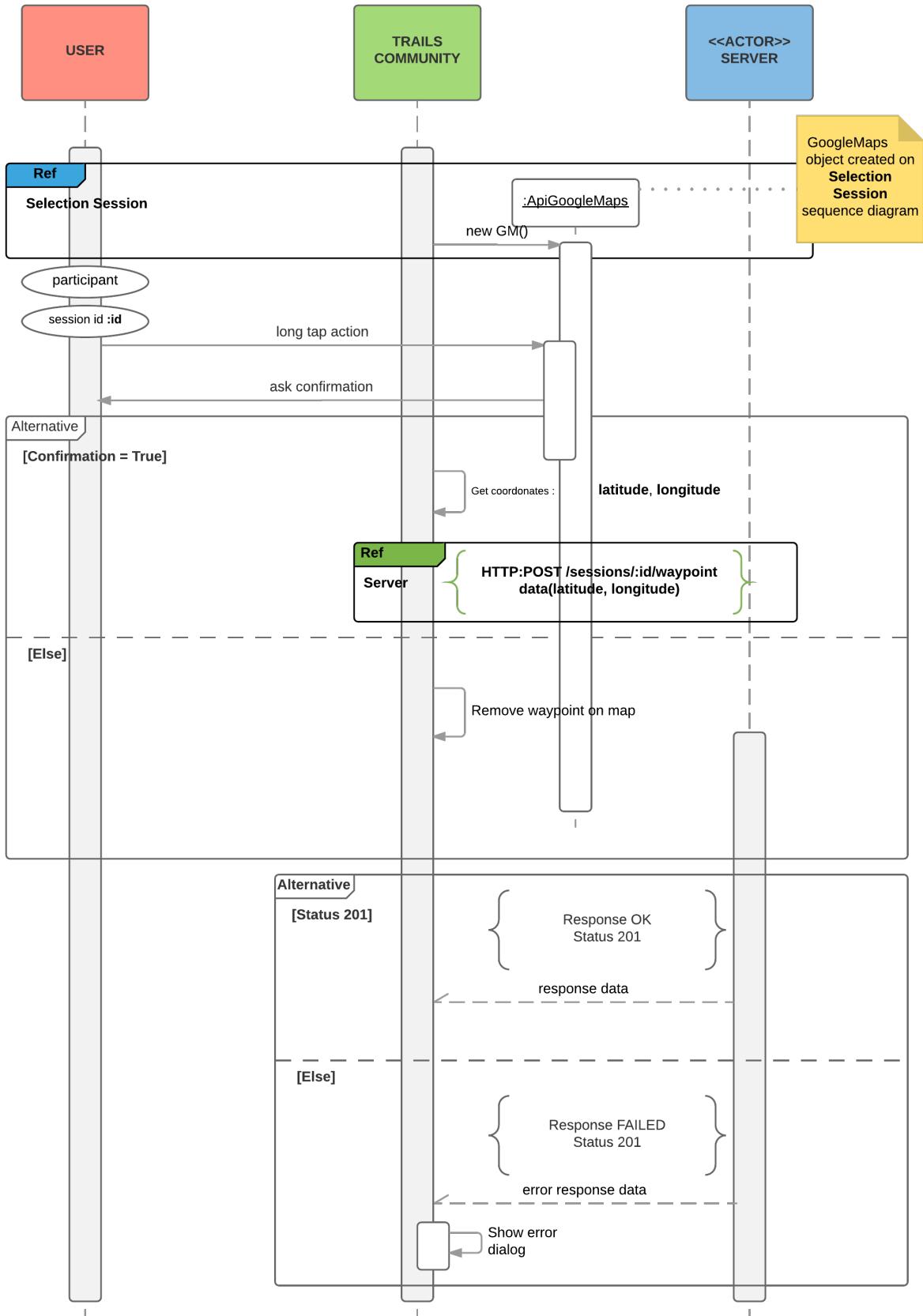


FIGURE 3.2 – Diagramme de séquence du chat

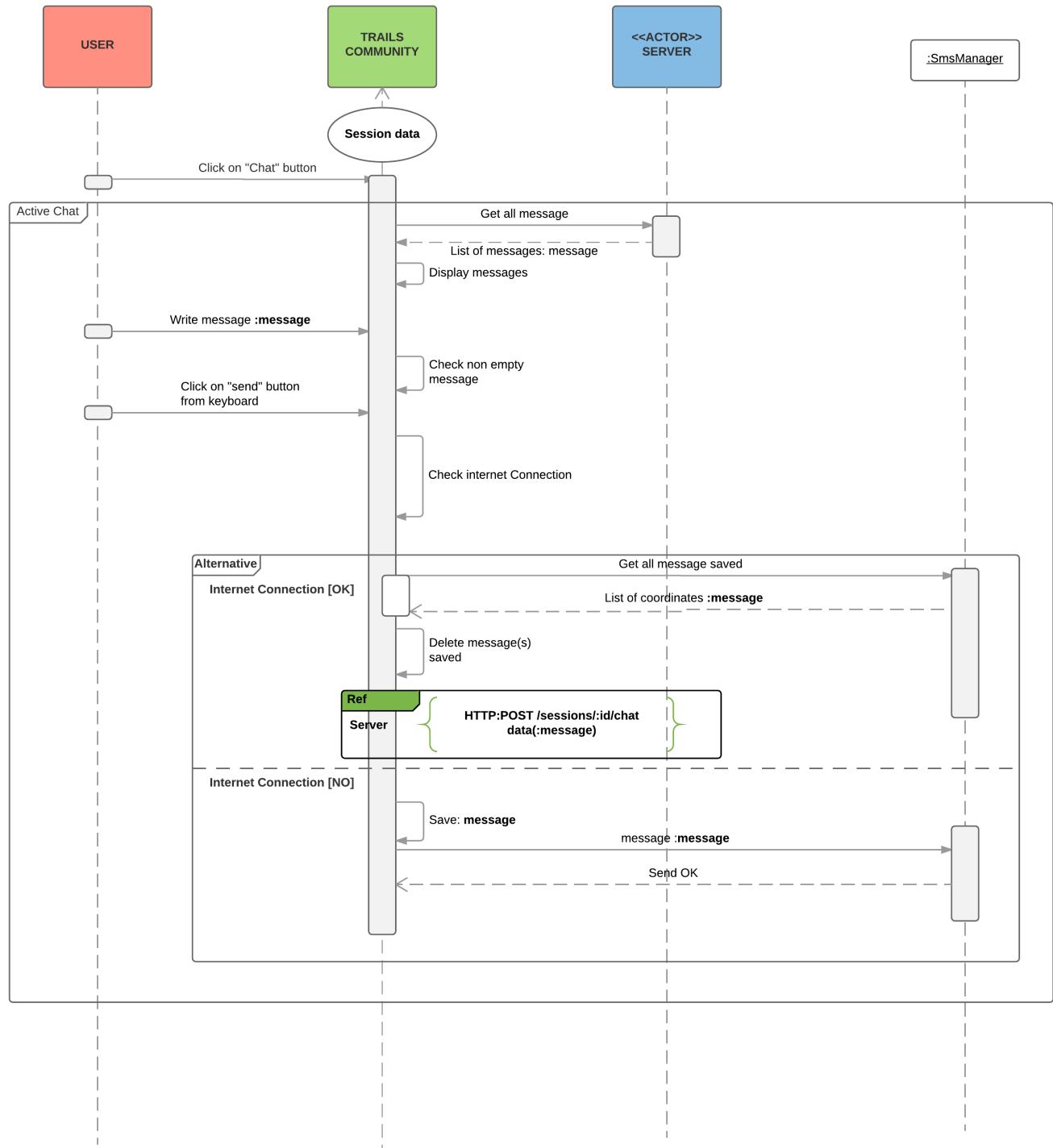


FIGURE 3.3 – Diagramme de séquence du la modification du profile

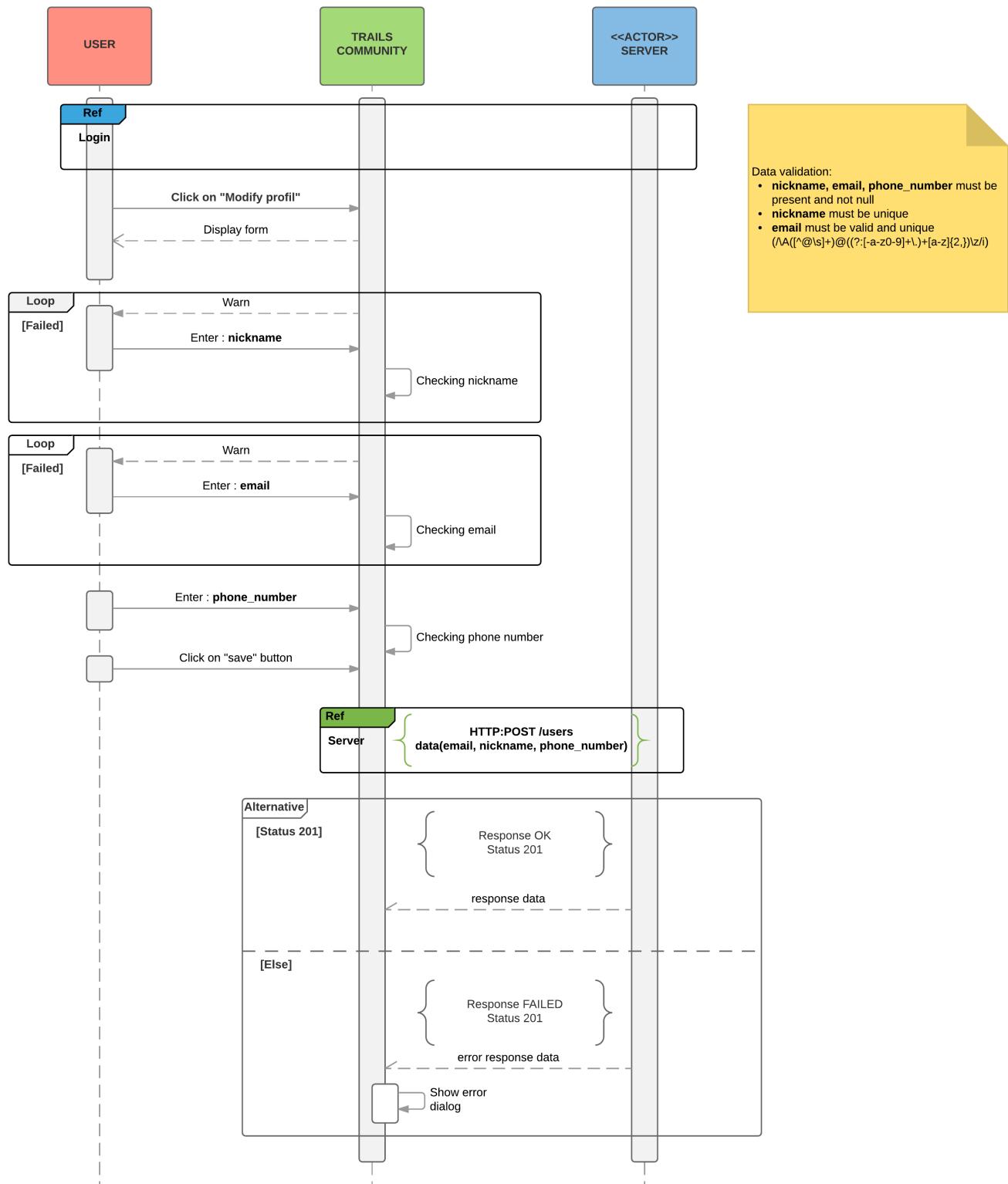


FIGURE 3.4 – Diagramme de séquence pour la sélection d'une session

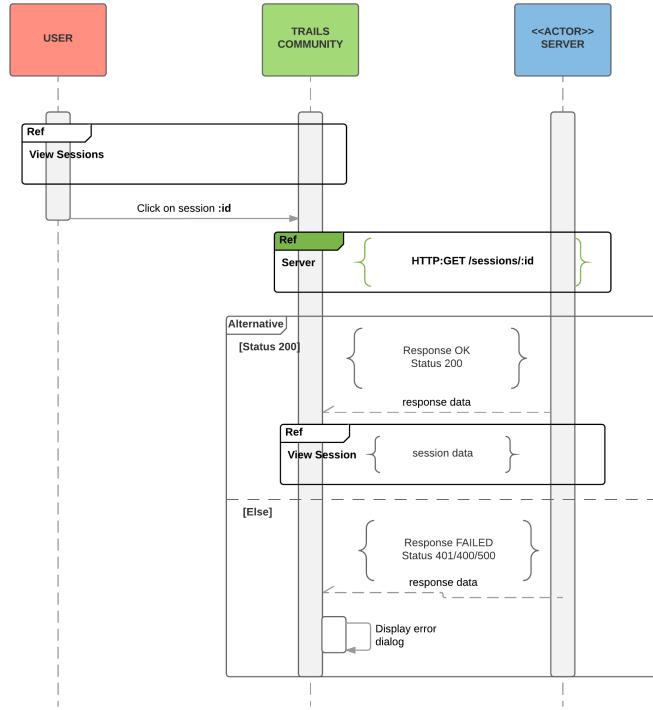


FIGURE 3.5 – Diagramme de séquence pour l'utilisation de la géolocalisation

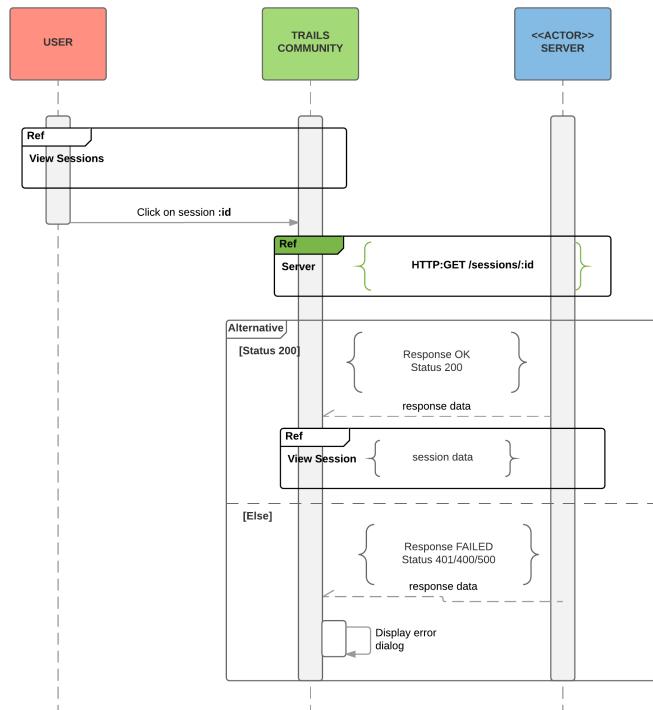
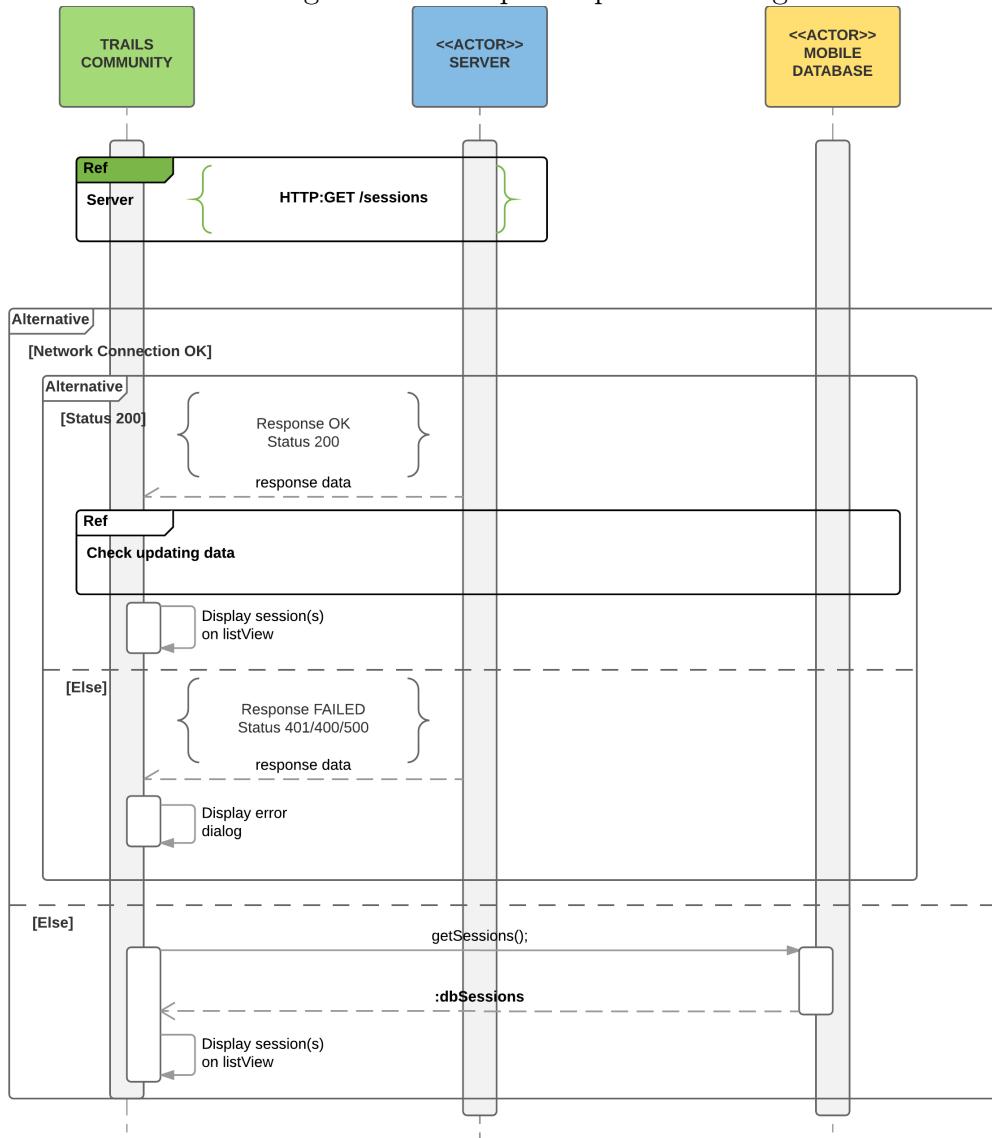


FIGURE 3.6 – Diagramme de séquence pour l'affichage de l'ensemble des sessions



Chapitre 4

Diagrammes d'activités

FIGURE 4.1 – Diagramme d'activité de l'ajout d'un waypoint

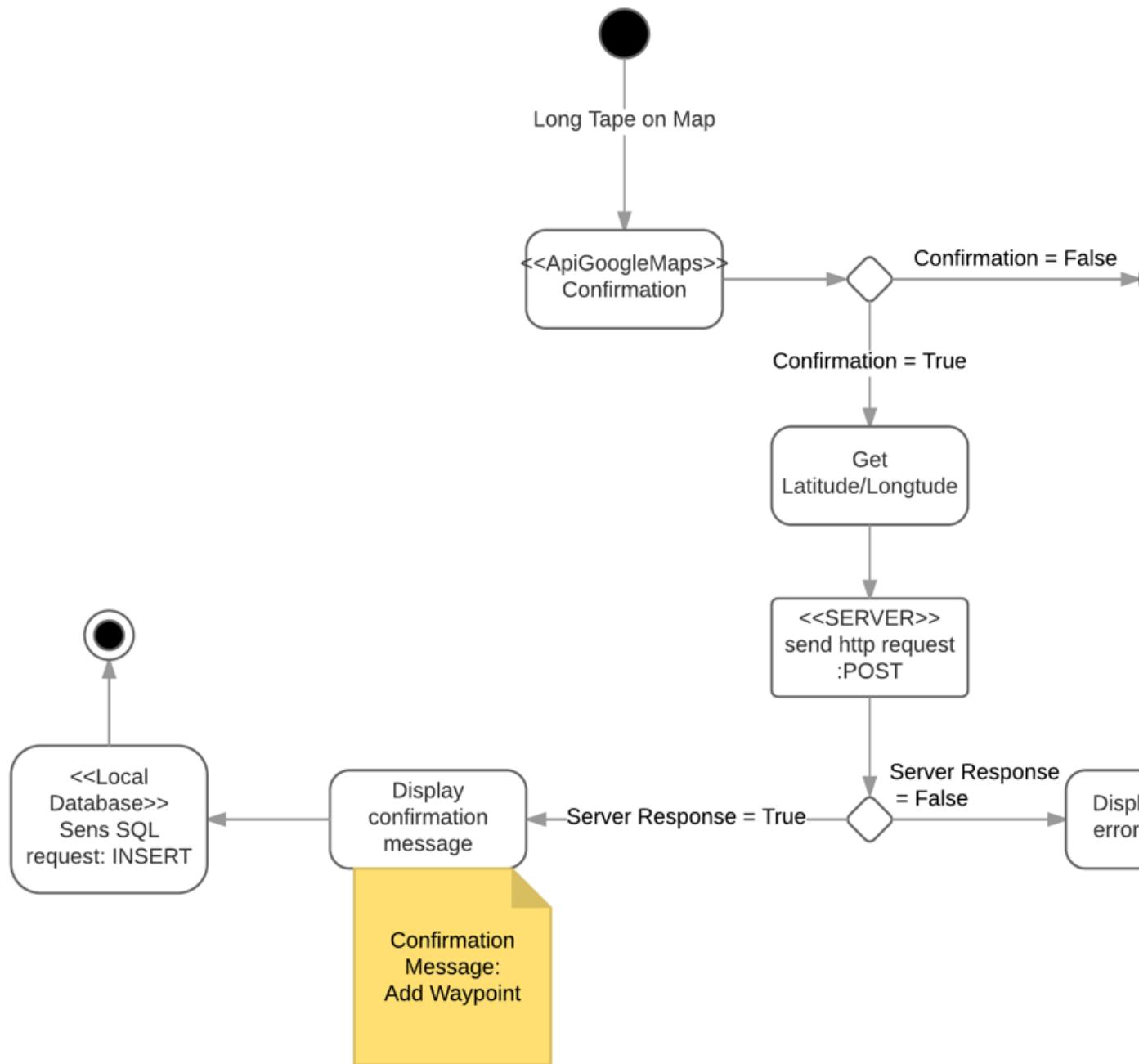


FIGURE 4.2 – Diagramme d'activité du chat

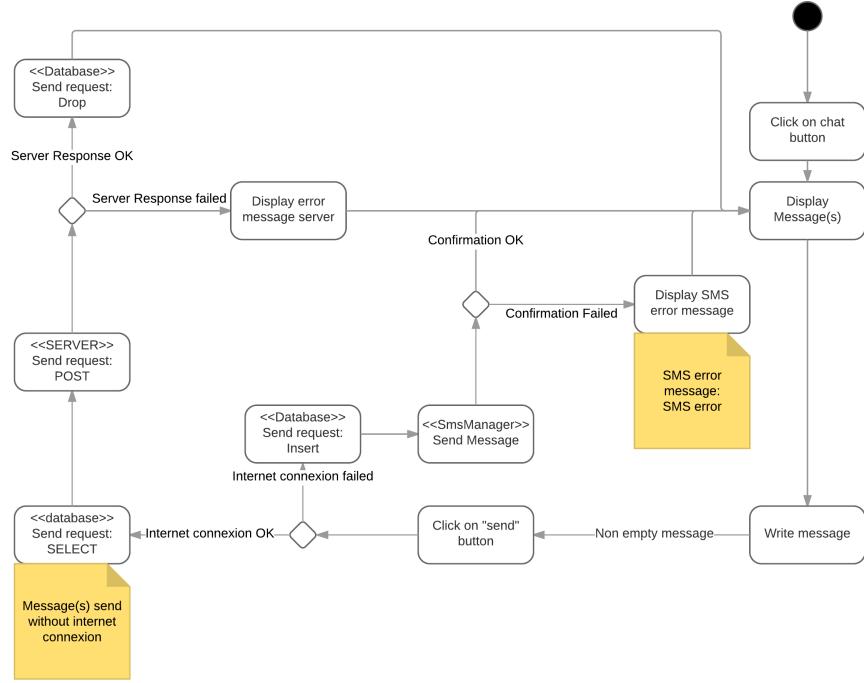


FIGURE 4.3 – Diagramme d'activité pour l'authentification

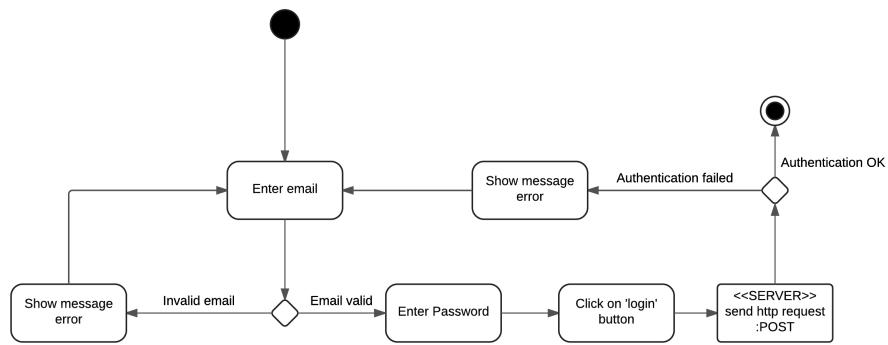


FIGURE 4.4 – Diagramme d’activité pour la modification du profile

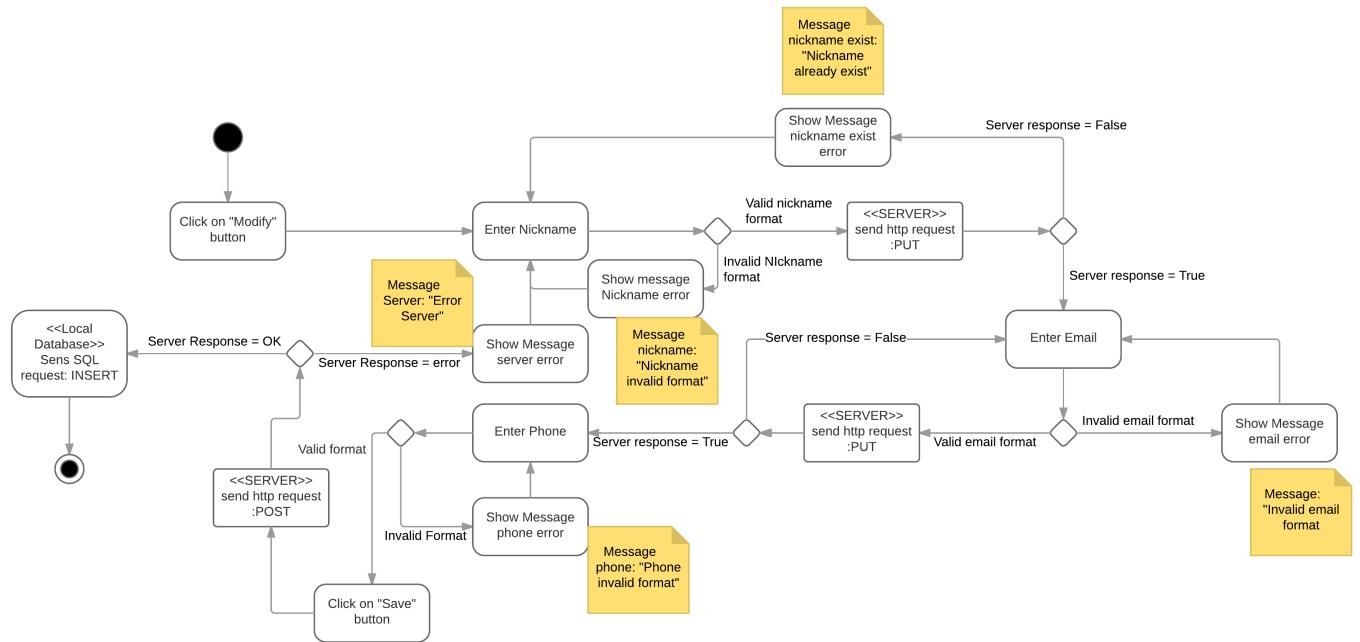


FIGURE 4.5 – Diagramme d’activité pour la sélection d’une session

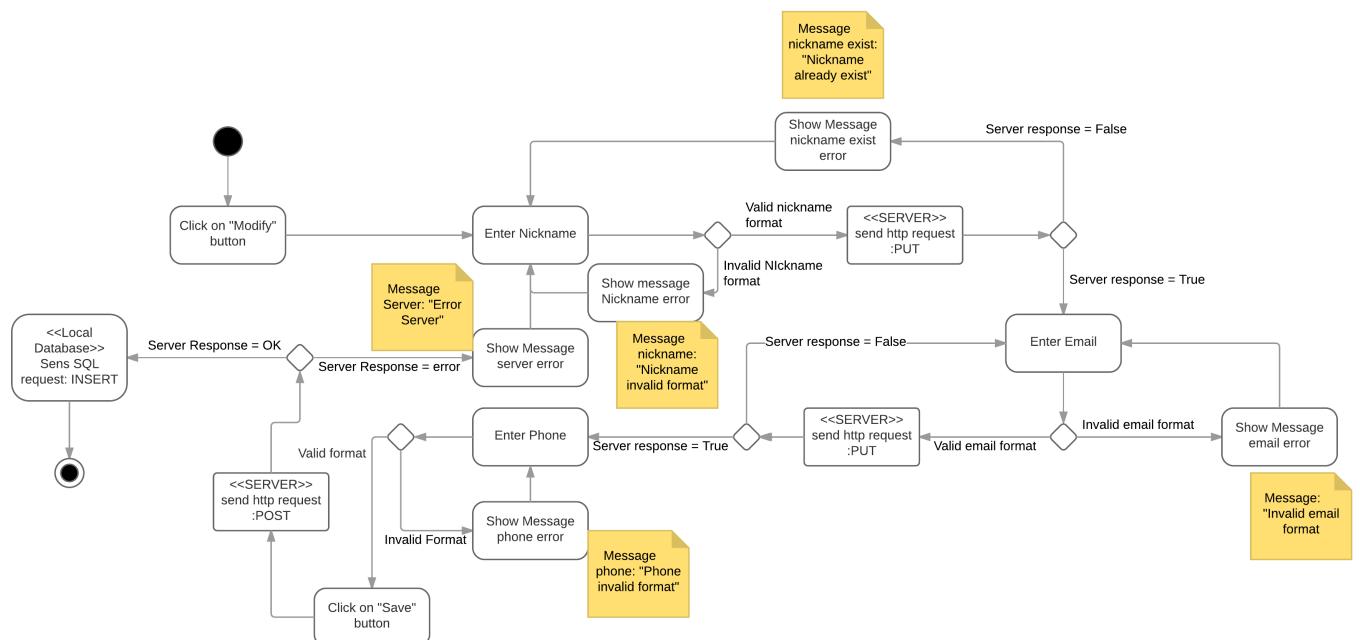


FIGURE 4.6 – Diagramme d’activité pour l’utilisation de la géolocalisation

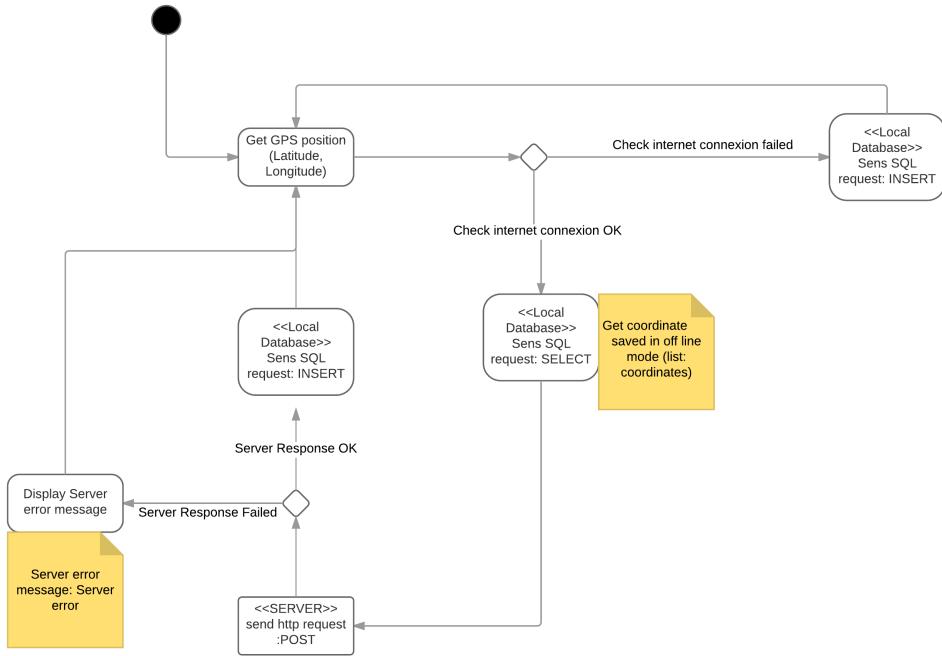
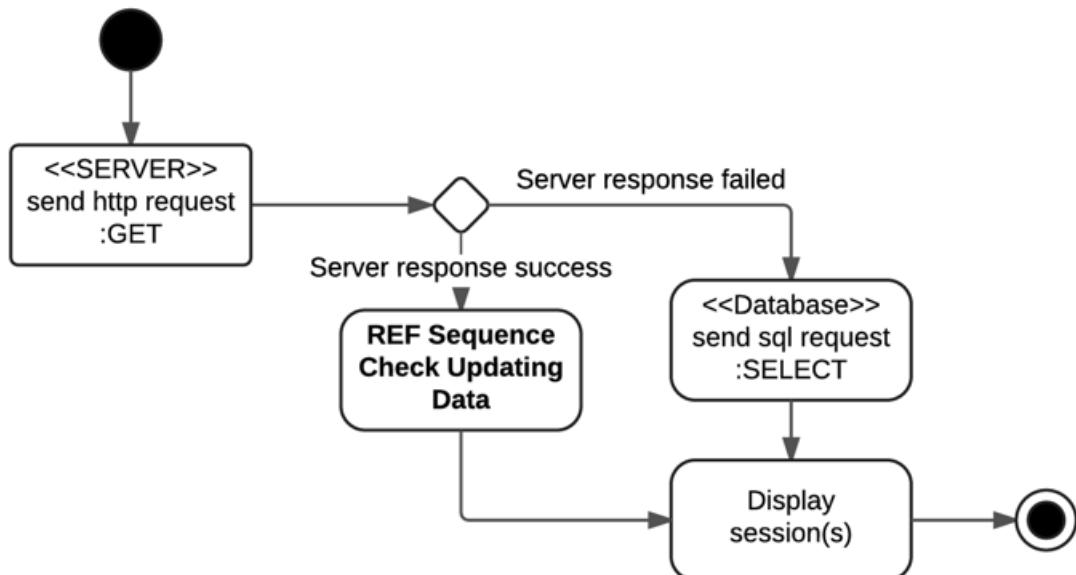


FIGURE 4.7 – Diagramme d’activité pour l’affichage de l’ensemble des sessions



Chapitre 5

Diagrammes d'états

FIGURE 5.1 – Diagramme d'état d'un acteur

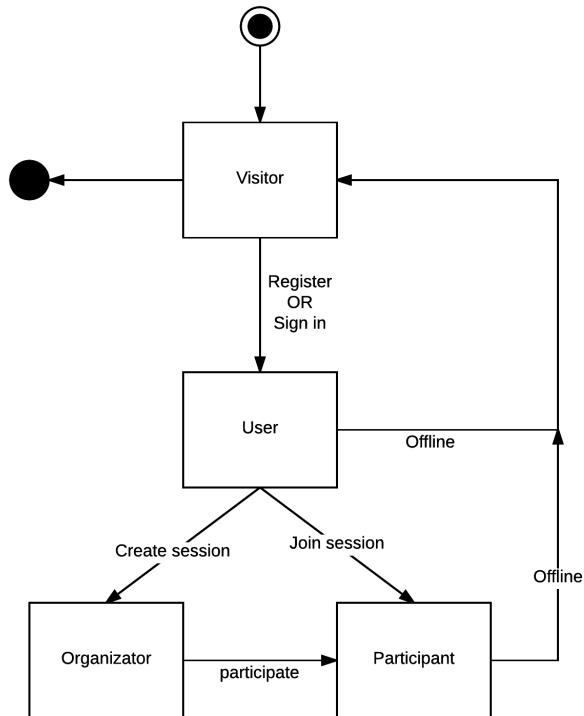


FIGURE 5.2 – Diagramme d'état d'une session

